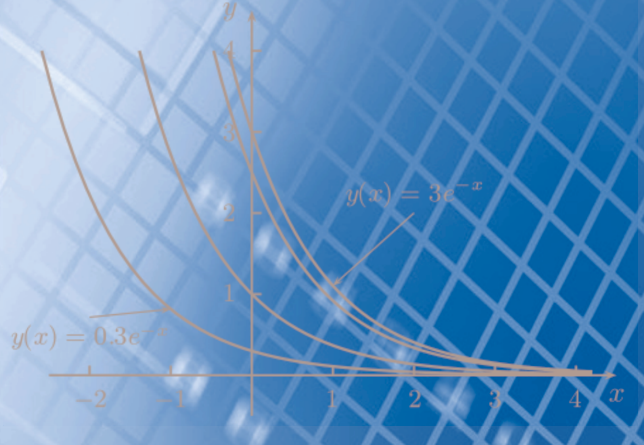
UNJBG

Solución Numérica de Ecuaciones Diferenciales Ordinarias y Parciales

Aplicaciones usando Excel y MATLAB



ISBN: 978-612-48189-4-3

Fondo Editorial

Edwin M. Pino Vargas

Solución Numérica de Ecuaciones Diferenciales Ordinarias y Parciales

Aplicaciones usando Excel y MATLAB

Edwin M. Pino Vargas

© Solución Numérica de Ecuaciones Diferenciales Ordinarias y Parciales
Aplicaciones Usando Excel y MATLAB

1.ª edición digital; Tacna, octubre de 2020
Libro disponible en: https://libros.unjbg.edu.pe

© 2020, Edwin M. Pino Vargas
ORCID del autor: 0000-0001-7432-4364

Dirección editorial: Av. Miraflores s/n (Tacna)

Revisión técnica: El presente libro cumplió con el sistema de evaluación por pares (doble ciego).
1.er evaluador: Dr. Ruben Ángel Agapito Ruiz
ORCID: 0000-0002-5819-259X
2.do evaluador: Dr. Jesús Abel Mejía Marcacuzco
ORCID: 0000-0002-9070-3898
3.cer evaluador: Dr. Eduardo Abraham Chávarri Velarde
ORCID: 0000-0002-8445-8996

Revisión de estilo: Paul Dorian Céspedes Sánchez
Diseño de portada y diagramación: José Luis Choque Dávila

ISBN: 978-612-48189-4-3
Depósito Legal
Hecho el depósito legal en la Biblioteca Nacional del Perú N° 2020-06985

UNJBG

Solución Numérica de Ecuaciones Diferenciales Ordinarias y Parciales

Aplicaciones usando Excel y MATLAB

Edwin M. Pino Vargas

Fondo Editorial

DEDICATORIA

A Dios, por permitirme vivir al lado de mis amores, que son la razón de mi vida: mi esposa Mely y mis adoradas hijas, Claudia y Diana.

A mis queridos padres, Fermín y Nélida, que supieron dar el rumbo adecuado a mi vida, con trabajo, dedicación, esmero y pasión por lo que hacemos en nuestras vidas.

A todos los miembros de mi familia, así como a mis amigos, quienes están pendientes de mis logros académicos y profesionales, y

A todos los hombres y mujeres que, desde los inicios de la vida en nuestro planeta, supieron relacionarse con el uso de las matemáticas para resolver los problemas cotidianos.

AGRADECIMIENTO

Un sincero agradecimiento a los árbitros revisores de este libro, de la Universidad Católica del Perú (PUCP) y de la Universidad Nacional Agraria La Molina (UNALM), por sus acertadas observaciones y recomendaciones en bien de mejorar el contenido de esta obra.

Al Vicerrectorado de Investigación, Instituto General de Investigación y Fondo Editorial de la Universidad Nacional Jorge Basadre Grohmann (UNJBG), por dar la rigurosidad necesaria para lograr que este material tenga la calidad deseada en términos de contenido y edición.

TABLA DE CONTENIDOS

Sobre el autor 15
Prólogo 17

Capítulo I: Fuentes de Error

1.1. Generalidades 21
1.2. Fuentes de Error 21
1.2.1. Error de Planteamiento 21
1.2.2. Error del Método 21
1.2.3. Error de Representación o Redondeo 22
1.2.4. Error de Operación o de Propagación 22
1.2.5. Error Residual 23
1.2.6. Error Numérico Total 23
1.2.7. Error Relativo 24

Capítulo II: Ecuaciones No Lineales

2.1. Soluciones Numéricas 27
2.2. Método de Newton-Raphson 28
2.2.1. Antecedentes Históricos 28
2.2.2. Descripción del Método 28
2.2.3. Formulación del Método 29
2.2.4. Convergencia y Valor Inicial 31
2.3. Estudio de Casos 32
2.3.1. Ecuación de Estado de Van Der Waals 32
2.3.2. Determinación del PH de un Ácido Monoprótico 36
2.3.3. Factor de Fricción en Tuberías 38
2.3.4. Datación Paleontológica 41
2.3.5. Estudio de Población 44
2.3.6. Vertedero de Francis 47
2.3.7. Flujo Uniforme en Canales 50
2.3.8. Deflexión de Vigas 53
2.3.9. Vibración Amortiguada 56
2.4. Problemas Propuestos 59

Capítulo III: Ecuaciones Diferenciales Ordinarias (EDO)

3.1. Generalidades 65
3.2. Uso de las EDO en las Ciencias e Ingeniería 66
3.3. Definiciones de las EDO 66
3.4. Solución de una EDO 67
3.5. Convergencia de una EDO 67
3.6. Métodos de Solución Numérica de las EDO 68
3.6.1. Método de Euler ("Predictor") 68
3.6.2. Método de Heun ("Predictor-Corrector") 69
3.6.3. Método de Runge-Kutta 1.er Orden 70
3.6.4. Método de Runge-Kutta 2.º Orden 72
3.6.5. Método de Runge-Kutta 3.er Orden 72
3.6.6. Método de Runge-Kutta 4.º Orden 72
3.6.7. Método de Runge-Kutta de Orden Superior 73
3.6.8. Método de Runge-Kutta Extendido 73
3.6.9. Método del Punto Medio 74
3.6.10. Método de Ralston 74
3.6.11. Métodos de Runge-Kutta-Fehlberg de 2.º Orden de Paso Variable 75
3.6.12. Métodos Multipaso 75
3.6.12.1. Método de Adams-Bashforth de 5.º Orden 75
3.6.12.2. Método de Adams-Bashforth de 4.º Orden 75
3.6.12.3. Método de Adams-Moulton de 5.º Orden 76
3.6.12.4. Método de Adams-Moulton de 4.º Orden 76
3.6.12.5. Método de Milne 77
3.6.13. ODE en MATLAB 77
3.7. Estudio de Casos 78
3.7.1. Tiempo de Vaciado de un Recipiente Cilíndrico Vertical 78
3.7.2. Tiempo de Vaciado de un Recipiente Esférico 84
3.7.3. Cálculo de un Eje Hidráulico o Curva de Remanso 91
3.7.4. Evolución de Temperatura en una Placa Calentada 96
3.7.5. Modelo de Crecimiento Poblacional de Malthus 98
3.7.6. Deflexión de una Viga en Voladizo 106
3.7.7. Movimiento Amortiguado 108
3.7.8. Velocidad de Caída de una Masa Esférica en Medio Acuoso 115
3.7.9. Modelo Lotka Volterra: Depredador–Presa 116
3.7.10. Balance de Masa en Estado no Estacionario 120
3.7.11. Concentración de Contaminantes 123
3.8. Problemas Propuestos 126

Capítulo IV: Ecuaciones Diferenciales Parciales (EDP)

4.1. Generalidades ... 135
4.2. Clasificación Matemática ... 136
4.3. Ecuaciones Elípticas ... 137
4.4. Ecuaciones Parabólicas ... 137
4.5. Ecuaciones Hiperbólicas ... 137
4.6. Métodos de Solución Numérica de las EDP ... 137
4.6.1. Ecuaciones Elípticas ... 137
4.6.2. Ecuaciones Parabólicas ... 138
4.6.2.1. Método Explícito ... 139
4.6.2.2. Métodos Implícitos ... 140
4.6.3. Ecuaciones Hiperbólicas ... 143
4.6.3.1. Método Explícito ... 144
4.6.3.2. Método Implícito ... 145
4.7. Estudio de Casos ... 146
4.7.1. Ecuación de Laplace para una Placa Calentada ... 146
4.7.2. Escurrimiento de un Fluido A través de un Medio Poroso ... 155
4.7.3. Ecuación de Conducción de Calor 1D ... 163
4.7.4. Flujo en Acuífero Libre con Método Crank-Nicolson ... 168
4.7.5. Consolidación Unidimensional en Suelos con un Estrato Compresible ... 176
4.7.6. Consolidación Unidimensional en Suelos con Varios Estratos Compresibles ... 184
4.8. Problemas Propuestos ... 192

Capítulo V: Aplicación Pdetool de MATLAB

5.1. Sobre PDETool de MATLAB ... 197
5.2. Cómo Utilizar PDETool de MATLAB ... 197
5.2.1. Estableciendo Dominios ... 197
5.2.2. Condiciones de Contorno ... 198
5.2.3. Especificaciones de la EDP a Resolver ... 198
5.2.4. Selección de Parámetros ... 199
5.2.5. Refinando la Malla ... 199
5.2.6. Solución de la EDP ... 199
5.3. Estudio de Casos ... 199
5.3.1. Líneas de Flujo o Corriente y Equipotenciales ... 199
5.3.2. Transferencia de Calor en una Placa Plana ... 201

5.3.3. Viga en Voladizo Sujeta a Carga Puntual 208
5.3.4. Flujo Debajo de una Presa Impermeable e Incorporación de un Dren 211
5.3.5. Perímetros de Protección de Pozos 215

Apéndice A: Introducción a MATLAB

6.1. Introducción 219
6.2. Aspectos Fundamentales del Lenguaje 219
6.3. Directorios 219
6.4. Definición de Variables y Operatoria Básica 220
6.4.1. Declaración de Matrices 221
6.4.2. Indexación de Elementos de una Matriz 221
6.4.3. Matrices Especiales 222
6.4.4. Operaciones con Matrices 222
6.5. M-Files 223
6.5.1. Creación de Funciones 223
6.5.2. Operadores Lógicos 224
6.5.3. Gráficos 225
6.6. Acerca de la Ayuda de MATLAB 229

Apéndice B: Introducción a GUIDE de MATLAB

7.1. Introducción 233
7.2. Diseño del GUIDE 233
7.3. Iniciando GUIDE 234
7.4. Flujo de Operación con GUI 237
7.5. Organización de los Objetos Gráficos 238
7.6. Partes de GUIDE 240
7.7. Propiedades de los Controles 240

REFERENCIAS 251

LISTA DE FIGURAS

Figura 2.1. Aproximación de Newton-Raphson 29
Figura 2.2. Ilustración de una iteración del método de Newton (la función *f* se muestra en azul y la línea de la tangente en negro) 30
Figura 2.3. Solución en hoja de cálculo Excel para la ecuación de estado de Van der Waals 33
Figura 2.4. Formulación en hoja de cálculo Excel para la ecuación de estado de Van der Waals 34
Figura 2.5. Resultados MATLAB para la ecuación de estado de Van der Waals 35
Figura 2.6. Resultado gráfico del proceso iterativo de la ecuación de estado de Van der Waals 35
Figura 2.7. Solución Excel para la determinación del pH de un ácido monoprótico 36
Figura 2.8. Resultado gráfico del proceso iterativo para la determinación del pH de un ácido monoprótico 37
Figura 2.9. Resultados MATLAB para la determinación del pH de un ácido monoprótico 38
Figura 2.10. Solución Excel para la determinación del factor de fricción *f* 39
Figura 2.11. Resultado gráfico del proceso iterativo para la determinación del factor de fricción *f* 40
Figura 2.12. Resultados MATLAB para la determinación del factor de fricción *f* 41
Figura 2.13. Solución Excel para la datación paleontológica 42
Figura 2.14. Resultado gráfico del proceso iterativo para la ecuación de la datación paleontológica 43
Figura 2.15. Resultados MATLAB para la determinación de la edad del fósil 44
Figura 2.16. Solución Excel para la determinación del tiempo de la función combinada de la población 45
Figura 2.17. Resultado gráfico del proceso iterativo para la ecuación combinada de la población 46
Figura 2.18. Resultados MATLAB para la ecuación combinada de la población 47
Figura 2.19. Elementos característicos de un vertedero rectangular en pared delgada 48
Figura 2.20. Solución Excel para la determinación del valor *H* del vertedero Francis 49
Figura 2.21. Resultado gráfico del proceso iterativo para el valor *H* del vertedero Francis 49
Figura 2.22. Resultados MATLAB para el tirante H del vertedero Francis 50
Figura 2.23. Solución Excel para la determinación del valor *H* de Manning 51
Figura 2.24. Resultado gráfico del proceso iterativo para el valor *H* de Manning 52
Figura 2.25. Resultados MATLAB para el tirante *H* de Manning 53
Figura 2.26. Solución Excel para la determinación del valor de la posición de la flecha máxima y el valor de esta 54
Figura 2.27. Resultado gráfico del proceso iterativo para el valor de la posición de la flecha máxima 55

Figura 2.28. Resultados MATLAB para la posición de la flecha máxima y el valor de esta 56
Figura 2.29. Solución Excel para la determinación del tiempo en el que se alcanza un desplazamiento de 4 unidades 57
Figura 2.30. Resultado gráfico del proceso iterativo para el valor del tiempo de desplazamiento 57
Figura 2.31. Resultados MATLAB para el tiempo de desplazamiento establecido 58
Figura 3.1. Procedimiento del método de Euler 68
Figura 3.2. Procedimiento del método de Heun 70
Figura 3.3. Recipiente cilíndrico vertical 78
Figura 3.4. Vaciado del cilindro vertical para la solución analítica 79
Figura 3.5. Vaciado del cilindro vertical para la solución de Euler 80
Figura 3.6. Salida del programa para el tiempo de vaciado del método Euler 81
Figura 3.7. Salida del programa para el tiempo de vaciado del método Euler mejorado 82
Figura 3.8. Salida del programa para el tiempo de vaciado del método Heun o *Predictor-Corrector* 83
Figura 3.9 Salida del programa para el tiempo de vaciado del método RK 23 84
Figura 3.10. Integración con Wolfram Mathematica, *online integrator* 86
Figura 3.11. Solución analítica para el tiempo de descarga en la parte inferior del recipiente esférico 86
Figura 3.12. Solución de Euler para el tiempo de descarga en la parte inferior del recipiente esférico 87
Figura 3.13. Solución de Euler en MATLAB para el tiempo de descarga de la parte inferior del recipiente esférico 88
Figura 3.14. Solución de Euler mejorado para el tiempo de descarga de la parte inferior del recipiente esférico 89
Figura 3.15. Solución de Heun para el tiempo de descarga de la parte inferior del recipiente esférico 90
Figura 3.16. Solución de Runge-Kutta 23 para el tiempo de descarga de la parte inferior del recipiente esférico 91
Figura 3.17. Solución Excel del método de Euler para la ecuación dinámica del flujo gradualmente variado 93
Figura 3.18. Solución MATLAB del método de Euler para la ecuación dinámica del flujo gradualmente variado 94
Figura 3.19. Solución MATLAB del método de Runge-Kutta para la ecuación dinámica del flujo gradualmente variado 95
Figura 3.20. Solución Excel para el problema de la placa calentada 96
Figura 3.21. Solución RK para el problema de la placa calentada 97
Figura 3.22. Modelo de crecimiento poblacional de Malthus 105
Figura 3.23. Solución Excel para la viga en voladizo 107
Figura 3.24. Solución MATLAB para la viga en voladizo 108

Figura 3.25. Solución Excel del movimiento sobre amortiguado usando RK extendido 110
Figura 3.26. Solución Excel del movimiento críticamente amortiguado usando RK extendido 110
Figura 3.27. Solución Excel del movimiento subamortiguado usando RK extendido 111
Figura 3.28. Solución MATLAB del movimiento sobre amortiguado usando RK extendido 114
Figura 3.29. Solución MATLAB del movimiento críticamente amortiguado usando RK extendido 114
Figura 3.30. Solución MATLAB del movimiento subamortiguado usando RK extendido 115
Figura 3.31. Resultados MATLAB para la velocidad de caída de una masa esférica en medio acuoso 116
Figura 3.32. Solución Excel para la evolución de la población Depredador-Presa 118
Figura 3.33. Solución MATLAB para la evolución de la población Depredador-Presa 119
Figura 3.34. Solución MATLAB "ode45" para la evolución de la población Depredador-Presa 120
Figura 3.35. Solución Excel para la ecuación de balance de masa en estado no estacionario 122
Figura 3.36. Solución ode45 para la ecuación de balance de masa en estado no estacionario 123
Figura 3.37. Integración con Wolfram Mathematica, *online integrator* 123
Figura 3.38. Solución Excel para el decaimiento de la concentración del contaminante 124
Figura 3.39. Solución Excel para el decaimiento de la concentración del contaminante usando paso de tiempo h=1 semana 124
Figura 3.40. Solución Excel para el decaimiento de la concentración del contaminante usando paso de tiempo h=10 semanas 125
Figura 3.41. Solución Excel para el decaimiento de la concentración del contaminante usando paso de tiempo h=15 semanas 125
Figura 3.42. Solución MATLAB para el decaimiento de la concentración del contaminante 126
Figura 4.1. Malla para la solución de EDP de dos variables 139
Figura 4.2. Célula computacional de EDP de método explícito 140
Figura 4.3. Malla explícita e implícita para la solución de EDP 141
Figura 4.4. Células computacionales del método implícito simple 143
Figura 4.5. Células computacionales del método implícito de Crank-Nicolson 143
Figura 4.6. Placa discretizada en 9 nudos y condiciones de borde o frontera 146
Figura 4.7. Sistema de ecuaciones en notación matricial 147
Figura 4.8. Sistema de ecuaciones en notación matricial completa 148
Figura 4.9. Sistema de ecuaciones en notación matricial, matriz inversa y matriz respuesta 148
Figura 4.10. Configuración Excel para proceso iterativo 149
Figura 4.11. Solución Excel de malla de 3 x 3 150

Figura 4.12. Solución Excel de malla de 10 x 10 151
Figura 4.13. Solución incluyendo el factor de relajación 152
Figura 4.14. Formulación en Excel para el proceso iterativo 152
Figura 4.15. Archivo "txt" de datos finales 153
Figura 4.16. Curvas de contorno obtenidas con el comando "contour" de MATLAB 154
Figura 4.17. Solución MATLAB para la placa calentada (2D) 155
Figura 4.18. Solución MATLAB para la placa calentada (3D) 155
Figura 4.19. Esquema de presa y volumen de control 156
Figura 4.20. Malla discretizada para el flujo debajo de una presa 157
Figura 4.21. Sistema de ecuaciones en notación matricial para la presa 158
Figura 4.22. Valores resultantes de cargas hidráulicas en los nudos de la malla 158
Figura 4.23. Opción de Excel para cálculo iterativo 159
Figura 4.24. Valores finales luego de proceso iterativo 159
Figura 4.25. Curvas de potenciales calculados 160
Figura 4.26. Campo de velocidades 161
Figura 4.27. Diagrama de distribución de presiones 162
Figura 4.28. Célula computacional de forma explícita 164
Figura 4.29. Malla de diferencias finitas 165
Figura 4.30. Resultados en Excel de temperatura en función del tiempo y espacio en una barra delgada 166
Figura 4.31. Resultados en MATLAB de temperatura en función del tiempo y espacio en una barra delgada 167
Figura 4.32. Esquema generalizado de la geometría de una excavación en acuífero libre 168
Figura 4.33. Representación del nivel piezométrico con geometría de acuífero confinado 169
Figura 4.34. Representación del nivel piezométrico con geometría de acuífero libre 169
Figura 4.35. Flujo de agua en una geometría de acuífero libre 169
Figura 4.36. Variables y parámetros de un esquema de excavación en terreno saturado y homogéneo 171
Figura 4.37. Discretización del problema en la dirección x 172
Figura 4.38. Discretización del problema parabólico 1D 172
Figura 4.39. Solución Excel para ecuación parabólica 1D que gobierna el flujo en acuífero libre hacia un dren 174
Figura 4.40. Curvas de nivel freático a 120 h y 240 h 174
Figura 4.41. Curvas de nivel freático obtenidas con MATLAB 176
Figura 4.42. Discretización del proceso unidimensional 178
Figura 4.43. Diagrama de exceso de presión versus profundidad, resultado Excel 180
Figura 4.44. Diagrama de flujo para la solución computacional del proceso de consolidación unidimensional para un estrato compresible 181
Figura 4.45. Curva de consolidación unidimensional para un estrato 183
Figura 4.46. Sistema de *Ns* estratos compresibles 184

Figura 4.47. Sistema de *Ns* estratos compresibles discretizados 185
Figura 4.48. Diagrama de flujo para la solución computacional del proceso de consolidación unidimensional con varios estratos compresibles 187
Figura 4.49. Curva de consolidación unidimensional para 4 estratos compresibles 191
Figura 5.1. Especificaciones para EDP 200
Figura 5.2. Flujo uniforme de izquierda a derecha 200
Figura 5.3. Flujo uniforme de izquierda a derecha incluyendo un obstáculo de sección rectangular 200
Figura 5.4. Flujo uniforme de izquierda a derecha incluyendo un obstáculo de sección circular 201
Figura 5.5. Flujo uniforme de izquierda a derecha incluyendo una fuente y un sumidero 201
Figura 5.6. Placa plana para transferencia de calor 202
Figura 5.7. Espacio de trabajo, elementos geométricos 202
Figura 5.8. Caja de diálogo para punto de inserción 203
Figura 5.9. Inserción de figuras geométricas, triángulo 203
Figura 5.10. Generando placa con agujero triangular 203
Figura 5.11. Condiciones de borde tipo Dirichlet 204
Figura 5.12. Condiciones de borde tipo Neumann 204
Figura 5.13. Condiciones de borde tipo Neumann-Convectiva 204
Figura 5.14. Condiciones de borde Dirichlet de T° constante 204
Figura 5.15. Caja de diálogo para especificar EDP 205
Figura 5.16. Refinado de malla 205
Figura 5.17. Caja de diálogo para definición de parámetros 206
Figura 5.18. Simulación, resultados en 2D 206
Figura 5.19. Simulación, resultados en 3D 206
Figura 5.20. Caja de diálogo para especificar la EDP parabólica 207
Figura 5.21. Caja de diálogo para especificar tiempos de simulación y valores iniciales 207
Figura 5.22. Parámetros para simulación en estado transitorio 207
Figura 5.23. Viga en voladizo 208
Figura 5.24. Definición de tipo y geometría para viga en voladizo 208
Figura 5.25. Condiciones de borde de viga en voladizo (1) 209
Figura 5.26. Condiciones de borde de viga en voladizo (2) 209
Figura 5.27. Condiciones de borde de viga en voladizo (3) 210
Figura 5.28. Parámetros y resultados de viga en voladizo 210
Figura 5.29. Presa impermeable con pantalla 211
Figura 5.30. Geometría de la presa 212
Figura 5.31. Refinado de la malla 212
Figura 5.32. Condiciones de borde 213
Figura 5.33. Revisión de "PDE Specification" 213
Figura 5.34. Resultados de simulación, flujo bajo la presa impermeable 213

Figura 5.35. Incorporación de un dren debajo de la presa 214
Figura 5.36. Detalle de la incorporación de un dren debajo de la presa 214
Figura 5.37. Simulación con la incorporación de un dren debajo de la presa a *Q*=20 214
Figura 5.38. Simulación con la incorporación de un dren debajo de la presa a *Q*=40 215
Figura 5.39. Simulación de un pozo de protección en un sistema subterráneo 215

Sobre el Autor

EDWIN M. PINO VARGAS, PH.D.
Doctoris Philosophiae en Recursos Hídricos
Magister of Scientiae en Recursos Hídricos
Ingeniero Agrícola
Ingeniero Civil

El autor, desde 1991, ha desarrollado, entre sus principales actividades: investigación, habiendo sido calificado como investigador por el Renacyt del Concytec a partir del año 2018; además, ha ejercido, hasta la fecha, la práctica profesional y la docencia universitaria en pregrado y posgrado (Maestría y Doctorado) en la especialidad de Ingeniería de Recursos Hídricos en las siguientes universidades: (1) Universidad Nacional Agraria La Molina (pregrado en los años 1993-1994). (2) En la actualidad, es profesor visitante del programa de doctorado en Recursos Hídricos de la Universidad Nacional Agraria La Molina. (3) Universidad Privada de Tacna (1996-2018). (4) Universidad Nacional Jorge Basadre Grohmann (1996-Actualidad).

También ha realizado actividad profesional como supervisor, proyectista y residente de estudios y obras en la especialidad de Ingeniería de Recursos Hídricos y afines (Modelación Numérica, Diseño de sistemas de riego presurizado, Hidrología e Hidrogeología, Hidráulica, Presas, Estructuras de protección de cauces, Defensas ribereñas, etc.).

En el ámbito internacional, se registra su participación como ponente en congresos latinoamericanos y mundiales de hidráulica desde el año 2011 como, por ejemplo, el Congreso Mundial de Hidráulica en China (2011), en Costa Rica (2013), en Holanda (2015), en Malasia (2017) y en Panamá (2019). Asimismo, como ponente en el Congreso Latinoamericano de Hidráulica en Santiago de Chile (2014), en Lima (2016), en Buenos Aires (2018), y con trabajo aceptado para el 2020 en Acapulco, México.

Finalmente, ha sido profesor titular, por más de 20 años, de la asignatura “Métodos Numéricos” en la Escuela de Ingeniería Geológica-Geotecnia de la Facultad de Ingeniería Civil, Arquitectura y Geotecnia en la Universidad Nacional Jorge Basadre Grohmann, asimismo, se ha desempeñado como profesor del curso de Métodos Numéricos, Aplicaciones en Ciencias e Ingeniería para profesores de otras universidades.

Prólogo

Normalmente, los procesos que se estudian en ciencias e ingeniería son representados por ecuaciones diferenciales ordinarias (EDO) o ecuaciones diferenciales parciales (EDP). En muchos casos, resolver una ecuación diferencial en forma analítica se convierte en una tarea muy complicada o casi imposible. Es así que los métodos numéricos cobran gran importancia, a pesar que una solución analítica sea exacta, los métodos numéricos sujetos a error son una alternativa viable. Una solución numérica es un valor aproximado de la solución, y aunque esa aproximación puede ser casi exacta, en muchos de estos métodos se puede desarrollar procesos iterativos hasta alcanzar la exactitud deseada, lo suficientemente exactos para satisfacer los requisitos del problema.

Este libro va dirigido a estudiantes de ciencias e ingeniería. Los requisitos mínimos son el cálculo diferencial e integral, así como la solución analítica de ecuaciones diferenciales ordinarias. Este libro busca, además, ilustrar en la solución numérica de las EDO y de las EDP, mediante aplicaciones en Excel y MATLAB. Está orientado a estudiantes con conocimiento en lenguajes de programación, específicamente MATLAB y GUIDE (entorno de programación visual disponible en MATLAB), que son un lenguaje y un entorno de programación bastante versátiles; pues proporcionan una serie de librerías y algoritmos elaborados que ayudan a facilitar las tareas de programación y que pueden usarse como herramientas de cálculo numérico, campo en el que son muy eficaces. Asimismo, se complementan con formulaciones en hoja de cálculo que llegan a ser una herramienta facilitadora para el entendimiento mecánico de la aplicación de los métodos empleados.

En este libro se expone la teoría, a veces, con justificaciones teóricas. Se presentan varios ejemplos resueltos aplicados a ciencias e ingeniería, exponiendo los algoritmos y la implementación de estos.

El Capítulo I está referido a fuentes de error, referidas a error de planteamiento, de método, de representación o redondeo, de operación o propagación, residual, numérico total y relativo.

El Capítulo II trata el tema de ecuaciones no lineales, específicamente enfocado al método de Newton-Raphson, un algoritmo para encontrar aproximaciones sucesivas de los ceros o raíces de una función real. Este algoritmo es el más usado y difundido por su facilidad de codificación; es eficiente en la solución de sistemas de ecuaciones no lineales, converge muy rápidamente y proporciona una muy buena precisión en los resultados.

El Capítulo III trata sobre ecuaciones diferenciales ordinarias, haciendo una descripción general, usos, definiciones y técnicas de solución. Se abordan métodos de solución muy simples como el de Euler y Heun hasta el desarrollo y aplicación de los métodos Runge-Kutta de primer orden hasta orden extendido y métodos multipaso. Aquí precisamos el uso de librerías MATLAB tipo ODE, en forma complementaria, y el estudio de varios casos típicos en ciencias e ingeniería.

En el Capítulo IV se desarrolla lo referido a las ecuaciones diferenciales parciales. La mayoría de los problemas en ciencias e ingeniería, de importancia práctica, están descritos por este tipo de ecuaciones diferenciales, y, fundamentalmente, por ecuaciones diferenciales de segundo orden. Se establecen soluciones a ecuaciones tipo elípticas, parabólicas e hiperbólicas, y sus correspondientes casos.

En el Capítulo V se hace referencia al uso de la caja de herramientas ("Toolbox" de MATLAB), que nos permite resolver ecuaciones diferenciales en derivadas parciales utilizando el método de elementos finitos. En particular, con la utilización de la interfaz gráfica denominada "PDETool" se especifica el dominio 2D de un problema, la triangulación del dominio, los coeficientes de la EDP y las condiciones de contorno, para luego obtener la solución del problema planteado.

Se ha incluido el Apéndice A sobre introducción a MATLAB y el Apéndice B sobre introducción a GUIDE de MATLAB. Se hace una reseña que pueda servir de base a los estudiantes que están iniciándose en programación. Asimismo, la guía GUIDE que se incluye puede permitir crear una interfaz gráfica, que es el vínculo entre el usuario y un programa computacional.

Finalmente, quiero agradecer al profesor de la Pontificia Universidad Católica del Perú, Rubén Agapito, Doctor of Philosophy in Mathematics (University Of California Santa Cruz), quien fue revisor de este libro, por sus valiosos aportes orientados a la mejora de esta obra. Asimismo, a mis estudiantes y colegas de la Universidad Nacional Jorge Basadre Grohmann por permitirme desarrollar la asignatura de Métodos Numéricos durante los últimos 25 años, años en los que se recopiló material que sirvió de base para escribir este libro.

Tacna, mayo de 2020

Edwin M. Pino Vargas

CAPÍTULO I
FUENTES DE ERROR

1.1. Generalidades

Los métodos numéricos son procedimientos matemáticos cuyo objetivo es la resolución numérica de problemas que carecen de expresión analítica para su resolución exacta (Chapra, 2017). Se expresan, en general, mediante algoritmos que especifican la secuencia de operaciones lógicas y aritméticas que conducen a la solución (normalmente aproximada) del problema planteado.

1.2. Fuentes de Error

Las fuentes de error en las matemáticas numéricas están ligadas a diferentes fuentes, como, por ejemplo: error de planteamiento, de método, de representación o redondeo, de operación o de propagación, residual, numérico total y relativo (Chapra et al., 2007; Chapra, 2017).

1.2.1. Error de Planteamiento

Normalmente, cuando se aborda un problema científico, el primer paso es construir un modelo matemático que lo represente; por ejemplo, si se quiere calcular el tiempo que tarda en llegar al suelo un objeto que se suelta desde una altura (h) utilizamos la ecuación del movimiento uniformemente acelerado: $h=gt^2/2$, siendo g la aceleración de la gravedad (Chapra, 2017).

En la construcción de un modelo matemático se asume una serie de simplificaciones (distribución uniforme de temperaturas, material homogéneo, isótropo, regiones esféricas, ausencia de fricción con el aire, etc.) que hacen que este, ya desde su mismo planteamiento, se aparte (esperamos que ligeramente) de la situación real observada. Además, en el modelo pueden intervenir constantes (como g en el ejemplo anterior) cuyo valor no se determina de manera exacta. Depende del científico, o del ingeniero, afinar, lo mejor posible, la elección del modelo matemático, de forma que la solución del mismo no se aparte significativamente de la solución del problema original.

1.2.2. Error del Método

Es el error debido a la propia forma en que está concebido el método numérico. En general, los métodos numéricos buscan dar soluciones aproximadas a los problemas, siendo el cálculo del error de aproximación una parte integrante del método. Un caso específico se refiere a la aproximación de funciones mediante su desarrollo en serie de potencias; por ejemplo, la función sen(x) puede aproximarse por su desarrollo de Taylor:

$$sen(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots + (-1)^{n+1}\frac{x^{2n-1}}{(2n-1)!}$$

La imposibilidad material de calcular los infinitos términos del desarrollo exige truncarlo con un número finito de términos, lo cual genera un tipo de error del método que, por su naturaleza, se denomina error de truncamiento. Por regla general, cuanto mayor sea el número de términos considerados, menor será el error de truncamiento.

1.2.3. Error de Representación o Redondeo

El ordenador también introduce errores. La capacidad limitada de representación del ordenador impide representar de forma exacta constantes matemáticas *(π, e)*, o números irracionales *($\sqrt{2}$, $\sqrt{3}$...)*, que tienen un número infinito de decimales. Aunque un número tiene infinitas cifras decimales, el ordenador solo permite representarlo con un número finito de cifras significativas. En consecuencia, se produce un error que denominaremos error de representación o de redondeo. En el caso particular de MATLAB, la representación interna de los números se hace con un total de 16 cifras decimales. Ello significa que los números reales, que solo se diferencien a partir de la décimo sexta cifra decimal, serán considerados iguales a los efectos de su tratamiento con este programa:

```
>> a=0.1234567891234567 89;

>> b=0.1234567891234567 88;

>> a-b

ans = 0
```

1.2.4. Error de Operación o de Propagación

El error de representación o de redondeo también afecta a las operaciones aritméticas que se realicen con el ordenador; el resultado de cada operación se redondea de acuerdo con el número máximo de decimales que puede representar el ordenador; de esta forma, en una sucesión de operaciones, se pueden ir produciendo sucesivos errores que se van arrastrando (propagando), lo que ocasiona una pérdida general de exactitud, puesto que se van perdiendo las cifras desechadas en cada paso.

Podemos ilustrar la importancia del error de propagación con un ejemplo muy conocido. Se trata del cálculo de la expresión $[(1+10^{-20}-1)x10^{20}]$. Esta expresión, operada en el orden que se indica, da, en la mayoría de los ordenadores, como resultado 0, en contraste con su resultado correcto que es 1.

La explicación, como es fácil de adivinar, está en la operación $1+10^{-20}$, donde, al tener los sumandos tanta diferencia en magnitud para la capacidad de representación del ordenador, su suma se redondea a 1, con las consiguientes consecuencias sobre el resto de las operaciones de la expresión, dando,

finalmente, el resultado de 0; por tanto, es importante tener en cuenta tanto el número de operaciones como el orden en que se realizan para evitar minimizar el impacto del error de propagación.

1.2.5. Error Residual

Muchos métodos numéricos requieren la repetición indefinida de un proceso de cálculo. Es cuando la expresión matemática del método especifica que la solución del problema se obtiene como límite de una sucesión, esto es:

$$\text{Solución} = \lim_{n\to\infty} \{x_i\}_{i=1}^{n}$$

La imposibilidad material de extender el proceso de cálculo indefinidamente obliga a limitar el número de términos de la sucesión a un valor finito, considerando que, para este valor, la solución aproximada alcanzada es un resultado suficientemente satisfactorio; por tanto, se produce otro tipo de error, que denominaremos *error residual*, que representa la diferencia debida a los términos no calculados.

1.2.6. Error Numérico Total

El error numérico total será la suma de todos los errores cometidos. Limitándonos a analizar los errores, desde el modelo matemático hasta el que se obtiene con la solución numérica, las dos fuentes principales de error y las medidas para atenuarlas (no eliminarlas) suelen ser:

- Reducir el error del método, principalmente el error de truncamiento. Para reducir este error se procede a la mejora de la exactitud del método numérico empleado, lo que suele acarrear una formulación más compleja y, generalmente, un incremento en el número de operaciones aritméticas, lo que a su vez hace aumentar el error de redondeo, además de incrementar el tiempo de cálculo.
- Reducir el error de redondeo. Como medidas para reducirlo, se presta atención al orden y a la forma en que se realizan las operaciones, y se incrementa el número de cifras significativas con las que opera el ordenador; por ejemplo, empleando números de doble precisión. En este último caso, esto supone aumentar las necesidades de memoria y almacenamiento del ordenador. En casos extremos, ello puede obligar a reformular completamente el método.

Como vemos, el comportamiento de los errores de truncamiento y redondeo responde, en general, a curvas con tendencias contrarias; por lo que, en muchas ocasiones, el remedio para un tipo de error incrementa el otro. Dado el caso, se hace necesario llegar a una situación de compromiso entre los dos tipos de error, lo que requiere cierto nivel de experiencia por parte del analista,

y muchas veces obliga a una serie de pruebas de ensayo-error hasta llegar a la mejor solución.

1.2.7. Error Relativo

En los problemas de cálculo numérico, si x es la solución (exacta) a un problema y x^* es la solución aproximada obtenida, el error absoluto cometido en la aproximación es simplemente $|x-x^*|$. En la práctica suele ser mucho más interesante calcular el error relativo. Es evidente que un error de una milésima apenas tiene importancia si el valor de x se mide en la escala de los millones; pero el mismo error de una milésima es muy grave si el valor de x se mide en la escala de las millonésimas.

El error relativo se define como:

$$\varepsilon_r = \frac{|x - x^*|}{|x|}$$

CAPÍTULO II
ECUACIONES NO LINEALES

2.1. Soluciones Numéricas

Hace mucho tiempo que hemos venido utilizando la fórmula cuadrática para resolver ecuaciones cuadráticas; asimismo, se presentan ecuaciones no lineales que requieren otro tipo de soluciones.

$$f(x) = ax^2 + bx + c = 0 \tag{2.1}$$

Aunque la fórmula cuadrática es útil para resolver las ecuaciones de la forma $f(x)$, existen muchas funciones en la ingeniería donde las raíces no se pueden determinar con tal facilidad. Existen, incluso, funciones que no se pueden resolver en forma analítica. En esta situación es que los métodos numéricos proporcionarán medios eficientes para obtener resultados. Es así que se han definido los métodos cerrados y abiertos. Los métodos cerrados se fundamentan en que una función cambia de signo en el intervalo que encierra la raíz o solución, y para desarrollar el algoritmo donde se encuentra la raíz se necesitan dos valores iniciales (límite inferior y límite superior), entre los cuales se encuentra la misma. Los métodos cerrados se clasifican en método de bisección y método de regla falsa.

Los métodos abiertos, a diferencia de los cerrados, calculan, en cada iteración, una aproximación a la raíz, y no requieren verificar si esta aproximación genera, o no, un intervalo que contenga una raíz. Los métodos abiertos son: método de punto fijo, método de Newton-Raphson, método de la secante y método de raíces múltiples. En este caso, nos vamos a referir específicamente al método de Newton-Raphson.

El método de Newton-Raphson requiere de la función y su derivada; cuando es relativamente fácil obtenerla no se tiene inconvenientes en usar este método. El problema surge cuando se hace muy complejo obtener la derivada, ante esto surgen otros métodos como el de la secante, que es un método para encontrar los ceros de una función de forma iterativa. La secante es una variación del método de Newton-Raphson, donde en vez de calcular la derivada de la función en el punto de estudio, teniendo en mente la definición de derivada, se aproxima la pendiente a la recta que une la función evaluada en el punto de estudio y en el punto de la iteración anterior. Este método es de especial interés cuando el costo computacional de derivar la función de estudio y evaluarla es demasiado elevado, por lo que el método de Newton-Raphson no resulta atractivo.

2.2. Método de Newton-Raphson

2.2.1. Antecedentes Históricos

En análisis numérico, el método de Newton (conocido también como el método de Newton-Raphson) es un algoritmo para encontrar aproximaciones sucesivas de los ceros o raíces de una función real (Deuflhard, 2004). Este método numérico fue descrito por *sir* Isaac Newton en *De analysi per aequationes numero terminorum infinitas* ('Sobre el análisis mediante ecuaciones con un número infinito de términos', escrito en 1669 y publicado en 1711 por William Jones) y en *De metodis fluxionum et serierum infinitarum* (escrito en 1671, traducido y publicado como *Método de las fluxiones* en 1736 por John Colson); sin embargo, su descripción difiere, en forma sustancial, de la descripción moderna presentada. Newton aplicaba el método solo a polinomios y no consideraba las aproximaciones sucesivas x_n, calculaba una secuencia de polinomios para llegar a la aproximación de la raíz x. Finalmente, Newton ve el método puramente algebraico y falla al no ver la conexión con el cálculo.

Isaac Newton probablemente derivó su método, de forma similar, aunque menos preciso que el método de François Viète. La esencia del método de Viète puede encontrarse en el trabajo del matemático persa Sharaf al-Din al-Tusi (Kelley, 2003). El método de Newton-Raphson es llamado así por el matemático inglés Joseph Raphson (contemporáneo de Newton), quien se hizo miembro de la Royal Society en 1691 por su libro *Aequationum Universalis*, publicado en 1690, que contenía este método para aproximar raíces. Newton, en su libro *Método de las fluxiones*, describe el mismo método en 1671, pero no fue publicado hasta 1736; lo que significa que Raphson había publicado este resultado 46 años antes y, aunque su trabajo no fue tan popular como los trabajos de Newton, fue reconocido posteriormente.

2.2.2. Descripción del Método

El método Newton-Raphson, quizá el más utilizado, es un método abierto en el sentido que no está garantizada su convergencia global. La única manera de alcanzar la convergencia es seleccionar un valor inicial lo suficientemente cercano a la raíz buscada. Así, se ha de comenzar la iteración con un valor razonablemente cercano al cero, denominado punto de arranque, valor inicial o valor supuesto (Ortega & Rheinboldt, 2000). La relativa cercanía del punto inicial a la raíz depende mucho de la naturaleza de la propia función; si esta presenta múltiples puntos de inflexión o pendientes grandes en el entorno de la raíz, entonces las probabilidades de que el algoritmo diverja aumentan, lo cual exige seleccionar un valor supuesto cercano a la raíz. Una vez hecho, el método linealiza la función por la recta tangente en ese valor supuesto. La

abscisa en el origen de dicha recta será, según el método, una mejor aproximación de la raíz que el valor anterior. Se realizarán sucesivas iteraciones hasta que el método haya convergido lo suficiente. El algoritmo de Newton-Raphson es deducido a partir de la pendiente señalada en la siguiente figura.

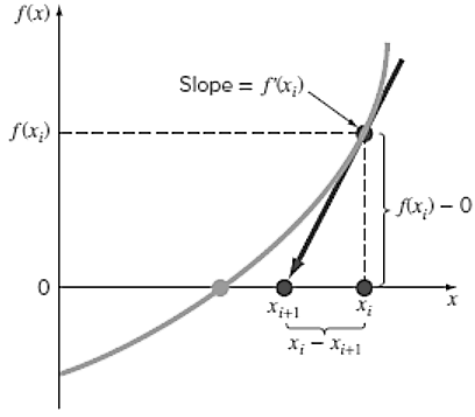


Figura 2.1.
Aproximación de Newton-Raphson

El método de Newton-Raphson se puede derivar de esta interpretación geométrica. En la figura mostrada, la primera derivada en x es equivalente a la pendiente:

$$f'(x_i) = \frac{f(x_i - 0)}{x_i - x_{i+1}}$$

Despejando x_{i+1}, se puede reorganizar para encontrar:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{2.2}$$

Esta ecuación es conocida como algoritmo de Newton-Raphson, donde $f'(x_i)$, representa la derivada de la función $f(x_i)$.

2.2.3. Formulación del Método

En la bibliografía existen tres formas de formular el algoritmo de Newton-Raphson. La primera se basa en una interpretación puramente geométrica basada en el método de la secante, pensando en que los puntos de iteración están lo suficientemente cerca; es decir, a una distancia infinitesimal, constituyéndose la secante por la tangente en un punto a la curva (Süli & Mayers, 2003). De esta forma, si por un punto de iteración trazamos la tangente a la curva, por extensión con el método de la secante, el nuevo punto

de iteración se tomará como la abscisa en el origen de la tangente (punto de corte de la tangente con el eje X). Esto es equivalente a linealizar la función, es decir, f se reemplaza por una recta tal que contiene al punto $[x_i, f(x_i)]$ y cuya pendiente coincide con la derivada de la función en el punto, $f'(x_i)$. La nueva aproximación a la raíz, x_{i+1}, se logra de la intersección de la función lineal con el eje X de abscisas. Matemáticamente se tiene:

$$f'(x_i) = \frac{f(x_i)}{x_i - x_{i+1}} \tag{2.3}$$

En la siguiente figura, se puede ver que x_{i+1} es una mejor aproximación que x_i para el cero (x) de la función f.

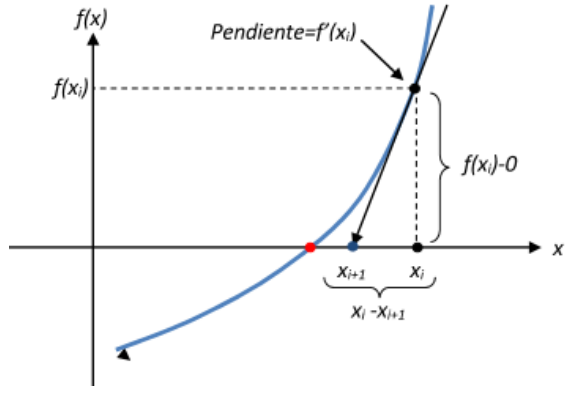


Figura 2.2.
Ilustración de una iteración del método de Newton
(la función f se muestra en azul y la línea de la tangente en negro)

Una segunda forma de obtener el algoritmo es desarrollando la función $f(x)$ en la serie de Taylor para un entorno del punto x_i.

$$f(x) = f(x_i) + f'(x_i)(x - x_i) + (x - x_i)^2 \frac{f''(x_i)}{2!} + \cdots \tag{2.4}$$

Si se trunca el desarrollo a partir del término de segundo grado y evaluamos en x_{i+1}; tenemos:

$$f(x_{i+1}) = f(x_i) + f'(x_{i+1})(x - x_i) \tag{2.5}$$

Si además se acepta que x_{i+1} tiende a la raíz, se ha de cumplir que $f(x_{i+1})=0$, luego, sustituyendo en la expresión anterior, obtenemos el algoritmo.

Finalmente, hay que indicar que el método de Newton-Raphson puede interpretarse como un método de iteración de punto fijo; así, dada la ecuación $f(x)=0$, se puede considerar como un método de iteración de punto fijo.

El pseudocódigo para el método se define de la siguiente manera; identificando: f es la función, f' la derivada de la función, $x0$ el valor inicial, tol la tolerancia o nivel de error, x_1 valor calculado en la iteración actual.

1. Entrar f, x_0, tol
2. Hacer $x1 = x_0 - f(x_0)/f'(x_0)$
3. Si $|x_0 - x1| <$ tol entonces escribir x_1
4. Fin
5. En caso contrario hacer $x_0 = x_1$ e ir a 2

2.2.4. Convergencia y Valor Inicial

Tenemos que ser conscientes que la convergencia no siempre va a suceder, especialmente cuando el valor inicial es muy alejado a la raíz. Para que esto ocurra, se tienen que dar una serie de condiciones sobre la función f, el punto de partida x_0 o la raíz x^*. En concreto, distinguiremos 3 tipos de convergencia (Aubanell et al., 1993):

- Local: Se dan condiciones sobre la raíz x^*.
- Semi local: Se dan condiciones sobre el punto de partida x_0.
- Global: Se dan condiciones sobre un intervalo.

Existe una gran cantidad de publicaciones con diversos resultados de convergencia para el método de Newton-Raphson (Bartle, 1976), (García y Nevot, 1997), (Díaz y Benítez, 1998). Una ventaja, sobre la elección del valor inicial próximo o cercano a la raíz buscada, es que, en ciencias e ingenierías, conocemos los procesos y podemos precisar con más certeza dicho valor. La manera de alcanzar la convergencia es seleccionar un valor inicial lo suficientemente cercano a la raíz buscada.

Se puede demostrar que el método de Newton-Raphson tiene convergencia cuadrática: si $''\alpha''$ es la raíz, entonces (Díaz et al., 1998), (Kharab & Guenther, 2006), (Llorente & Pérez, 2007), (Neuhauser & Torres, 2011):

$$|x_{k+1} - \alpha| \leq C\,|x_k - \alpha|^2 \qquad (2.6)$$

Para una cierta constante (C), esto significa que si en algún momento el error es menor o igual a 0.1, a cada nueva iteración doblamos (aproximadamente) el número de decimales exactos. En la práctica puede servir para hacer una

estimación aproximada del error. El error relativo entre dos aproximaciones sucesivas es (Ortega & Rheinboldt, 2000), (Grau & Noguera, 2001):

$$E = C\,\frac{|x_{k+1} - x_k|}{|x_{k+1}|} \tag{2.7}$$

De esta forma, se toma el error relativo como si la última aproximación fuera el valor exacto. Se detiene el proceso iterativo cuando este error relativo es aproximadamente menor que una cantidad preestablecida.

2.3. Estudio de Casos

Se presenta, a continuación, un conjunto de casos seleccionados en los que se pretende exponer la metodología adoptada para la solución de problemas en las ciencias e ingeniería, relacionados a la solución de las ecuaciones no lineales. Se expondrá la formulación física, matemática y la solución numérica haciendo uso de herramientas computacionales (Excel y MATLAB).

2.3.1. Ecuación de Estado de Van der Waals

Formulación del caso:

La ecuación de estado de Van der Waals (Duque-Vega & Gracia-Fadrique, 2015) para un gas real queda representada según:

$$\left(P + \frac{a}{V^2}\right)(V - b) = RT \tag{2.8}$$

Donde:

p: Presión del gas en atm.
T: Temperatura en °K.
R: Constante universal de los gases equivalente a 0.08205 atm-L/gmol-°K.
V: Volumen molar del gas en L/gmol, y
a, b: Constantes particulares para los gases.

Calcular el volumen molar V a 80 °C para una presión p=10 atm, para el gas CO_2, cuyas constantes a=3.599 y b=0.04267.

Solución:

La ecuación para un gas real, al multiplicarla por V^2 y reordenando sus términos se obtiene:

$$PV^3 - (Pb + RT)V^2 + aV - ab = 0 \tag{2.9}$$

De esta manera la ecuación anterior es no lineal de la forma $f(V)=0$, la cual puede ser resuelta usando los métodos de punto fijo, Newton-Raphson, secante, posición falsa, bisección, etc. En este caso, usaremos el algoritmo de Newton-Raphson donde F representa la función.

A continuación, procedemos a efectuar el cálculo de su derivada D a partir de F.

$$F = PV^3 - (Pb + RT)V^2 + aV - ab \tag{2.10}$$

$$D = 3PV^2 - 2(Pb + RT)V + a \tag{2.11}$$

Una vez obtenida la función de estado reordenada y su respectiva derivada, el algoritmo de Newton-Raphson puede ser escrito de la siguiente manera:

$$V_{i+1} = V_i - \left(\frac{F(V_i)}{D(V_i)}\right) \tag{2.12}$$

A continuación, establecemos una formulación en hoja de cálculo Excel para resolver el proceso iterativo hasta lograr el valor V deseado a un nivel de error preestablecido. Es importante imponer el valor inicial para el inicio del proceso iterativo, por lo tanto, podemos fijar como punto de partida $Vo=RT/p$, es decir, valores conocidos R de la constante de los gases, T la temperatura y p la presión.

	A	B	C	D	E	F
1	**Datos de Entrada**					
2	P=	10	atm			
3	T=	353.2	°K			
4	R=	0.08205	atm-L/gmol-°K			
5	a=	3.59900				
6	b=	0.04267				
7	**Funcion y Derivada**					
8	F	= PV^3-(Pb+RT)V^2+aV-ab				
9	D	= 3PV^2-2(Pb+RT)V+a				
10	Vi+1	= Vi - F(Vi)/D(Vi)				
11	**Valores Iniciales**					
12	Vo	= RT/P =	2.89801	Error =	0.00001	
13	**Solucion Usando Newton Raphson**					
14	**i**	**Vi**	**F(Vi)**	**D(Vi)**	**F(Vi)/D(Vi)**	**Vi+1**
15	0	2.89801	6.69274	85.11023	0.07864	2.81937
16	1	2.81937	0.35090	76.24733	0.00460	2.81477
17	2	2.81477	0.00117	75.74012	0.00002	2.81475
18	3	2.81475	0.00000	75.73842	0.00000	2.81475
19	4	2.81475	0.00000	75.73842	0.00000	2.81475

Figura 2.3.
Solución en hoja de cálculo Excel para la ecuación de estado de Van der Waals

Asimismo, mostramos la formulación Excel haciendo referencia al contenido de las celdas; es decir, se pueden observar las fórmulas ingresadas para lograr, luego de 5 iteraciones, el valor requerido. De esta forma podemos concluir

que el valor del volumen molar de CO2, V para una presión de 10 atm y 80 °C de temperatura es de 2.81475 L/gmol. Cabe destacar que, solo cambiando los contenidos de las celdas B2 a B6, podemos hacer cálculos adicionales variando presión, temperatura o tipo de gas.

	A	B	C	D	E	F
1	**Datos**					
2	P=	10	atm			
3	T=	=80+273.2	°K			
4	R=	0.08205	atm-L/gmol-°K			
5	a=	3.599				
6	b=	0.04267				
7	**Funci**					
8	F	$= PV^3-(Pb+RT)V^2+aV-ab$				
9	D	$= 3PV^2-2(Pb+RT)V+a$				
10	Vi+1	= Vi - F(Vi)/D(Vi)				
11	**Valore**					
12	Vo	= RT/P =	=+B4*B3/B2		Error =	0.00001
13	**Soluci**					
14	**i**	**Vi**	**F(Vi)**	**D(Vi)**	**F(Vi)/D(Vi)**	**Vi+1**
15	0	=+C12	=+$B$2*B15^3-($B$2*$B$6+$B$4*$B$3)*B15^2+$B$5*B15-$B$5*$B$6	=3*($B$2*B15^2)-2*($B$2*$B$6+$B$4*$B$3)*B15+$B$5	=+C15/D15	=+B15-E15
16	1	=+F15	=+$B$2*B16^3-($B$2*$B$6+$B$4*$B$3)*B16^2+$B$5*B16-$B$5*$B$6	=3*($B$2*B16^2)-2*($B$2*$B$6+$B$4*$B$3)*B16+$B$5	=+C16/D16	=+B16-E16
17	2	=+F16	=+$B$2*B17^3-($B$2*$B$6+$B$4*$B$3)*B17^2+$B$5*B17-$B$5*$B$6	=3*($B$2*B17^2)-2*($B$2*$B$6+$B$4*$B$3)*B17+$B$5	=+C17/D17	=+B17-E17
18	3	=+F17	=+$B$2*B18^3-($B$2*$B$6+$B$4*$B$3)*B18^2+$B$5*B18-$B$5*$B$6	=3*($B$2*B18^2)-2*($B$2*$B$6+$B$4*$B$3)*B18+$B$5	=+C18/D18	=+B18-E18
19	4	=+F18	=+$B$2*B19^3-($B$2*$B$6+$B$4*$B$3)*B19^2+$B$5*B19-$B$5*$B$6	=3*($B$2*B19^2)-2*($B$2*$B$6+$B$4*$B$3)*B19+$B$5	=+C19/D19	=+B19-E19

Figura 2.4.
Formulación en hoja de cálculo Excel para la ecuación de estado de Van der Waals

A continuación, planteamos una solución diferente, establecemos un código computacional en MATLAB, para la solución del algoritmo de Newton-Raphson para la ecuación de estado Van der Waals.

```
clc, clear;
%Datos de entrada
P=10;
T=353.2;
R=0.08205;
a=3.599;
b=0.04267;
%Error asignado para proceso de cálculo e iniciación
E=0.0001;
V=R*T/P;
d=0.01;
while abs(d)>E
F=P*V^3-(P*b+R*T)*V^2+a*V-a*b;
D=2*P*V^2-2*(P*b+R*T)*V+a;
d=F/D;
V1=V-d;
V=V1;
end
Volumen=V
```

Este programa, codificado en MATLAB, arroja como resultado: Volumen = 2.8147. En la siguiente figura se muestra una gráfica del proceso iterativo hasta la convergencia hacia la solución.

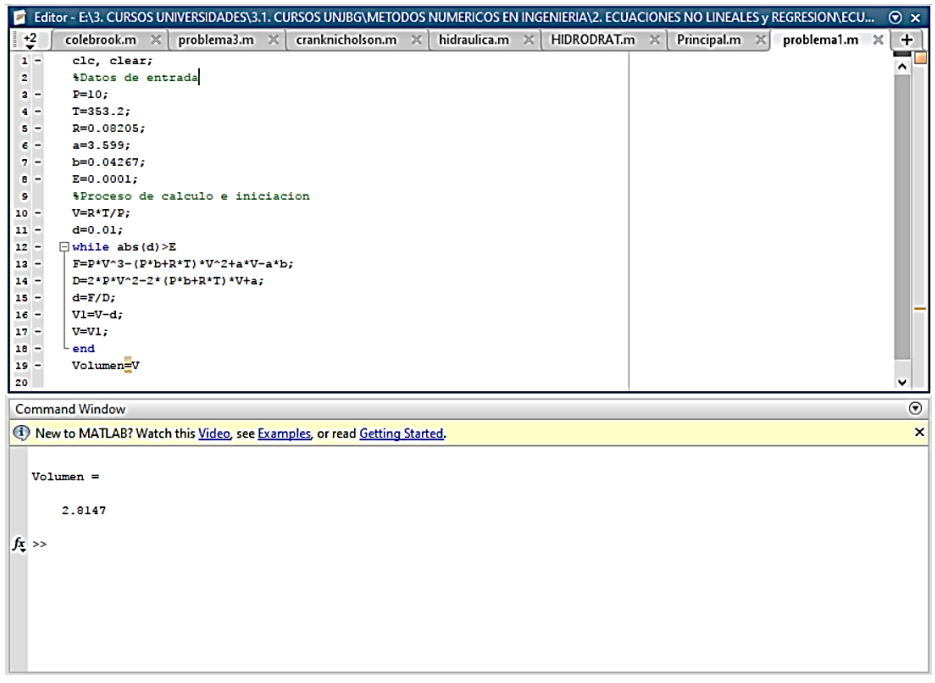


Figura 2.5.
Resultados MATLAB para la ecuación de estado de Van der Waals

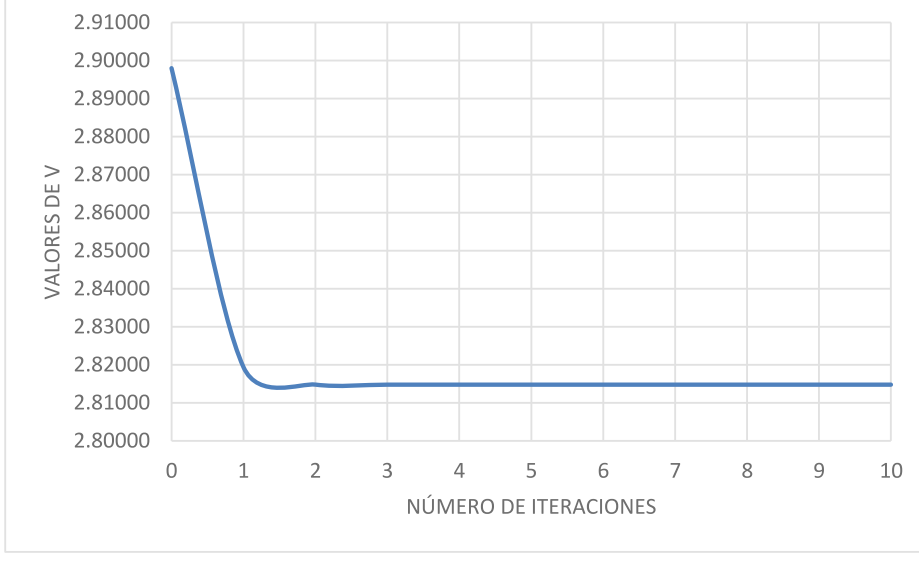


Figura 2.6.
Resultado gráfico del proceso iterativo de la ecuación de estado de Van der Waals

2.3.2. Determinación del pH de un Ácido Monoprótico

Formulación del caso:

En la determinación del pH de un ácido monoprótico se llega a la siguiente ecuación:

$$\frac{V_b}{V_a+V_b}C_b+[H^+]=\frac{K_w}{[H^+]}+\frac{V_a}{V_a+V_b}C_a\frac{1}{1+\frac{[H^+]}{K_a}} \tag{2.13}$$

Con volúmenes V_a=10 mL, V_b=3 mL, concentraciones C_a=C_b=0,1M, y constantes de equilibrio $K_a=10^{-3}$, $K_w=10^{-14}$. Se quiere determinar el pH de este ácido. Ayuda para no-químicos: pH = – log([H^+]). [H^+] es la incógnita (puede sustituirse por "H" para emplear una notación más familiar).

Empleando el método de Newton-Raphson con $[H^+]_0$ = 0.001, como valor inicial, calcular el valor pH.

Solución:

$$F=\frac{V_b}{V_a+V_b}C_b+[H^+]-\frac{K_w}{[H^+]}-\frac{V_a}{V_a+V_b}C_a\frac{1}{1+\frac{[H^+]}{K_a}} \tag{2.14}$$

$$D=1+\frac{K_w}{[H^+]^2}+\frac{V_a}{V_a+V_b}C_a\frac{1/K_a}{\left(1+\frac{[H^+]}{K_a}\right)^2} \tag{2.15}$$

La formulación en hoja de cálculo Excel permite resolver el proceso iterativo hasta lograr el valor Hi deseado a un nivel de error preestablecido.

	A	B	C	D	E	F
1	**Datos de Entrada**					
2	Va=	10	mL			
3	Vb=	3	mL			
4	Ca=Cb	0.10000	M			
5	Ka=	1.00E-03				
6	Kw=	1.00E-14				
7	Hi+1	= Hi - F(Hi)/D(Hi)				
8	**Valores Iniciales**					
9	Ho=	0.001		Error =	0.00001	
10	**Solucion Usando Newton Raphson**					
11	**i**	**Hi**	**F(Hi)**	**D(Hi)**	**F(Hi)/D(Hi)**	**Hi+1**
12	0	0.00100	-0.01438	20.23077	-0.00071	0.001711
13	1	0.00171	-0.00359	11.46620	-0.00031	0.002024
14	2	0.00202	-0.00034	9.41305	-0.00004	0.002060
15	3	0.00206	0.00000	9.21641	0.00000	0.002060
16	4	0.00206	0.00000	9.21434	0.00000	0.002060
17	5	0.00206	0.00000	9.21434	0.00000	0.002060
18	6	0.00206	0.00000	9.21434	0.00000	0.002060
19	7	0.00206	0.00000	9.21434	0.00000	0.002060
20	8	0.00206	0.00000	9.21434	0.00000	0.002060
21	9	0.00206	0.00000	9.21434	0.00000	0.002060
22	10	0.00206	0.00000	9.21434	0.00000	0.002060

Figura 2.7.
Solución Excel para la determinación del pH de un ácido monoprótico

La formulación de la celda C12 a la F12 se describe a continuación:

```
C12: +($B$3/($B$2+$B$3))*$B$4+B12-($B$6/B12)-.....
.....($B$2*$B$4/($B$2+$B$3))*(1/(1+(B12/$B$5))))
D12: =1+($B$6/B12^2)+(($B$2/($B$2+$B$3))*$B$4)*((1/$B$5)/(1+B12/$B$5)^2)
E12: =+C12/D12
F12: =+B12-E12
```

De esta manera obtenemos pH=-log(0.002060)=2.69. En la siguiente figura se observa una gráfica del proceso iterativo hasta la convergencia.

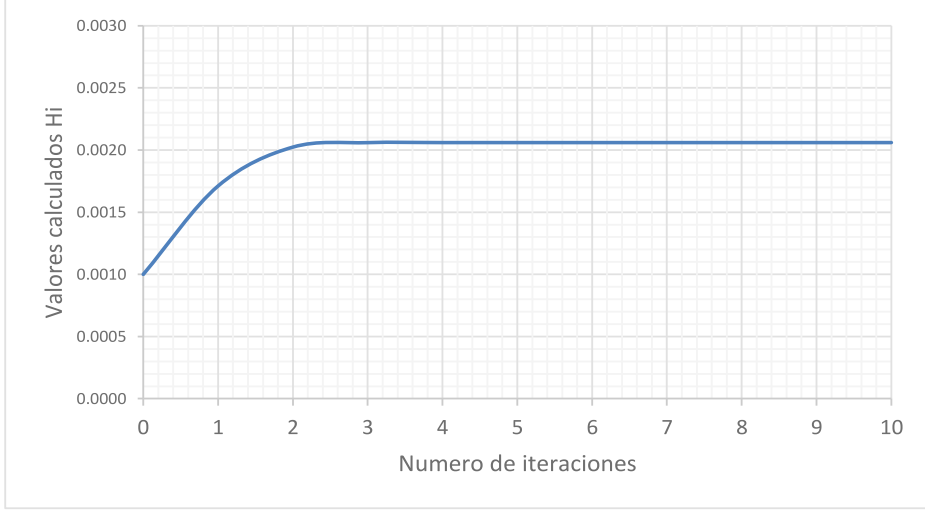


Figura 2.8.
Resultado gráfico del proceso iterativo para la determinación del pH de un ácido monoprótico

Asimismo, planteamos una solución con código computacional en MATLAB, para la solución del algoritmo de Newton-Raphson.

```
clc, clear;
%Datos de entrada
Va=10;
Vb=3;
Ca=0.10000;
Cb=Ca;
Ka=1.00E-03;
Kw=1.00E-14;
E=0.00001;
%Proceso de cálculo e iniciación
H=0.001;
d=0.01;
while abs(d)>E
F = ( V b / ( V a + V b ) ) * C b + H - ( K w / H ) -
```

```
(Va/(Va+Vb))*Ca*(1/(1+(H/Ka)));
D=1+(Kw/H^2)+(Va/(Va+Vb))*Ca*(1/Ka)/(1+(H/Ka))^2;
d=F/D;
H1=H-d;
H=H1;
end
Hidrogeno=H;
pH=-log10(H)
```

El programa codificado en MATLAB arroja como resultado: pH = 2.6861.

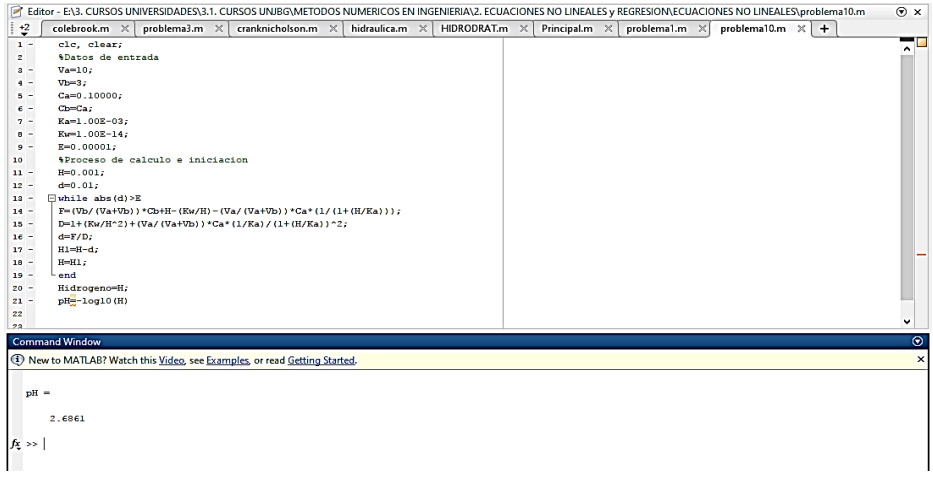


Figura 2.9.
Resultados MATLAB para la determinación del pH de un ácido monoprótico

2.3.3. Factor de Fricción en Tuberías

Formulación del caso:

Para el cálculo de las pérdidas por fricción en tuberías es mejor el empleo del método de solución del gradiente junto a las ecuaciones de Darcy-Weisbach y Colebrook-White, en vez del empleo de la ecuación de Hazen Williams y su respectivo coeficiente de rugosidad. El valor de rugosidad absoluta considerada es de 1.5×10^{-6} m y viscosidad del fluido de 1.14×10^{-6} m^2/s.

La expresión para el cálculo de pérdidas de carga en tuberías a presión y su respectivo factor de fricción f, se estable en las siguientes ecuaciones.

$$h_f = f\frac{L}{D}\left(\frac{V^2}{2g}\right) = \left(\frac{8fL}{\pi^2 g D^5}\right)Q^2 \tag{2.16}$$

$$\frac{1}{\sqrt{f}} = -2log_{10}\left(\frac{\varepsilon}{3.7D} + \frac{2.51}{Re\sqrt{f}}\right) \tag{2.17}$$

Esta ecuación se conoce como "ecuación de Darcy-Weisbach" y permite calcular las pérdidas de carga por fricción, donde f es el factor de fricción, L la longitud de la tubería, D el diámetro y Q el caudal. Para el cálculo del factor de fricción (f) en tuberías lisas o rugosas en régimen turbulento; es decir, número de Reynolds mayor a 4000, se utilizará la ecuación de Colebrook-White, donde ε es la rugosidad absoluta del material del que está fabricada la tubería (Pino et al., 2017).

Empleando el método de Newton-Raphson con $f_0 = 0.01$, como valor inicial, calcular el valor del factor de fricción f.

Solución:

$$F = \frac{1}{\sqrt{f}} + 2log_{10}\left(\frac{\varepsilon}{3.7D} + \frac{2.51}{Re\sqrt{f}}\right) \tag{2.18}$$

$$D = -\frac{1}{2}\left(\frac{1}{\sqrt{f}}\right)^3\left(1 + \frac{5.02}{ln(10)\,Re\left(\frac{\varepsilon}{3.7D} + \frac{2.51}{Re\sqrt{f}}\right)}\right) \tag{2.19}$$

La formulación en hoja de cálculo Excel permite resolver el proceso iterativo hasta lograr el valor fi deseado a un nivel de error preestablecido.

	A	B	C	D	E	F
1	**Datos de Entrada**					
2	Re=	308405				
3	ε=	0.000001522	m			
4	D=	0.15220	m			
5	f_{i+1}	= f_i - F(f_i)/D(f_i)				
6	**Valores Iniciales**					
7	f_o=	0.001			Error =	0.00001
8	**Solucion Usando Newton Raphson**					
9	**i**	**f_i**	**F(f_i)**	**D(f_i)**	**F(f_i)/D(f_i)**	**f_{i+1}**
10	0	0.00100	24.45296	-16241.16949	-0.00151	0.002506
11	1	0.00251	12.41409	-4157.05514	-0.00299	0.005492
12	2	0.00549	5.59646	-1305.71595	-0.00429	0.009778
13	3	0.00978	1.97180	-560.12720	-0.00352	0.013298
14	4	0.01330	0.40161	-357.49807	-0.00112	0.014422
15	5	0.01442	0.02312	-317.65893	-0.00007	0.014494
16	6	0.01449	0.00008	-315.33852	0.00000	0.014495
17	7	0.01449	0.00000	-315.33002	0.00000	0.014495
18	8	0.01449	0.00000	-315.33002	0.00000	0.014495
19	9	0.01449	0.00000	-315.33002	0.00000	0.014495
20	10	0.01449	0.00000	-315.33002	0.00000	0.014495

Figura 2.10.
Solución Excel para la determinación del factor de fricción f

De esta manera obtenemos el factor de fricción f=0.014495. En la siguiente figura se observa el proceso iterativo hasta conseguir la convergencia hacia la solución. La formulación de las celdas C10 a F10 es:

```
C10: =+(1/B10^0.5)+2*LOG10((2.51/($B$2*B10^0.5))+$B$3/(3.7*$B$4))
D10: =-0.5*((1/B10^0.5)^3)*(1+(2*2.51/((LN(10))*$B$2* ……..
…….. ((2.51/($B$2*B10^0.5))+($B$3/(3.7*$B$4))))))
E10: =+C10/D10
F10: =+B10-E10
```

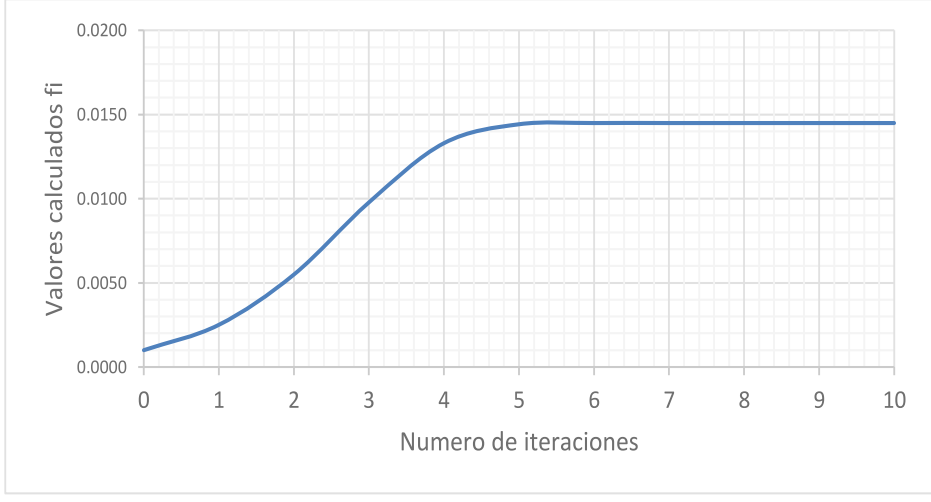


Figura 2.11.
Resultado gráfico del proceso iterativo para la determinación del factor de fricción f

De la misma manera, establecemos un código computacional en MATLAB para la solución del algoritmo de Newton-Raphson para determinar el factor de fricción de la ecuación Colebrook-White.

```
clc, clear;
%Datos de entrada
Re=308405;
e=0.000001522;
D=0.15220;
E=0.00001;
%Proceso de cálculo e iniciación
f=0.001;
d=0.01;
while abs(d)>E
F=(1/f^0.5)+2*log10((2.51/(Re*f^0.5))+e/(3.7*D));
D                                    =                                    -
0.5*((1/f^0.5)^3)*(1+(2*2.51/((log(10))*Re*((2.51/(Re*f^0.5))+(e/(3.7*
D))))));
d=F/D;
```

```
f1=f-d;
f=f1;
end
Friccion=f
```

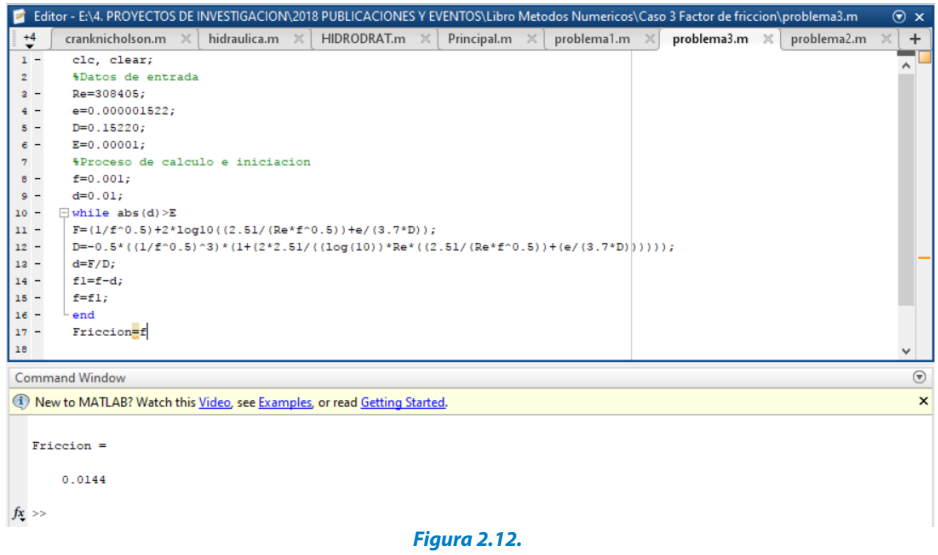


Figura 2.12.
Resultados MATLAB para la determinación del factor de fricción f

2.3.4. Datación Paleontológica

Formulación del caso:

En una muestra de suelo se pretende datar la aparición de ciertos organismos fósiles. Se conoce la cantidad de un cierto isótopo p, habitualmente C14, inicialmente presente antes de que el organismo falleciera, siendo la cantidad inicial $\mathbf{N}_O^{C14}$.A partir de ese momento comienza la desintegración isotópica, con una tasa marcada por la constante de desintegración λ. Si la cantidad medida de C14 en el momento actual es $N^{C14}(ta)$, la ecuación que rige la concentración en moles del C14 para un aporte externo es:

$$F(t) = N^{C14}(ta) - N_0^{C14} e^{-\lambda t} + \frac{v}{\lambda}\left(e^{-\lambda t} - 1\right) \qquad (2.20)$$

Donde para el isotopo C14: $\lambda = 0.00012378\ moles/año$, y $v = 10^{-8} moles/año$

Sabiendo que $N^{C14}(ta)$=0.0001 moles y $\mathbf{N}_O^{C14} = 1\ mol$ ¿Cuál será la edad del fósil hallado?

Solución:

En primer lugar, debemos establecer la forma de la función y su respectiva derivada para la concentración de moles del C14 por aportes externos, siendo esta:

$$F(t) = N^{C14}(ta) - N_0^{C14}e^{-\lambda t} + \frac{\upsilon}{\lambda}\left(e^{-\lambda t} - 1\right) = 0 \tag{2.21}$$

$$D(t) = \lambda N_0^{C14}e^{-\lambda t} - \frac{\upsilon}{\lambda}\left(\lambda e^{-\lambda t}\right) = 0 \tag{2.22}$$

La formulación en hoja de cálculo Excel permite resolver el proceso iterativo de Newton-Raphson, hasta lograr el valor *ti* deseado a un nivel de error preestablecido. Se muestran 10 iteraciones.

	A	B	C	D	E	F
1	**Datos de Entrada**					
2	λ=	0.00012378	moles/año			
3	υ=	0.00000001	moles/año			
4	$N^{C14}(ta)$=	0.00010	moles			
5	N_o^{C14}=	1.00000	moles			
6	t_{i+1}	= t_i - $F(t_i)/D(t_i)$				
7	**Valores Iniciales**					
8	t_o=	50000.000		Error =	0.00001	
9	**Solucion Usando Newton Raphson**					
10	**i**	**t_i**	**$F(t_i)$**	**$D(t_i)$**	**$F(t_i)/D(t_i)$**	**t_{i+1}**
11	0	50000.00000	-0.00204	0.000000255	-8003.37103	58003.37
12	1	58003.37103	-0.00074	0.000000095	-7875.65829	65879.03
13	2	65879.02931	-0.00027	0.000000036	-7540.42650	73419.46
14	3	73419.45581	-0.00009	0.000000014	-6710.08829	80129.54
15	4	80129.54410	-0.00003	0.000000006	-4938.98134	85068.53
16	5	85068.52544	-0.00001	0.000000003	-2293.62552	87362.15
17	6	87362.15096	0.00000	0.000000003	-395.08586	87757.24
18	7	87757.23682	0.00000	0.000000002	-10.12395	87767.36
19	8	87767.36077	0.00000	0.000000002	-0.01001	87767.37
20	9	87767.37078	0.00000	0.000000002	0.00000	87767.37
21	10	87767.37078	0.00000	0.000000002	0.00000	87767.37

Figura 2.13.
Solución Excel para la datación paleontológica

Con esta formulación obtenemos la edad del fósil hallado *t*=87 767.37 años. En la siguiente figura se observa el proceso iterativo hasta la convergencia hacia la solución. La formulación de las celdas C11 a F11 es:

```
C11: =+$B$4-$B$5*2.7173^(-$B$2*B11)+($B$3/$B$2)*(-1+2.7173^(-$B$2*B11))
D11: =+$B$2*$B$5*2.7173^(-$B$2*B11)-($B$3/$B$2)*($B$2*2.7173^(-$B$2*B11))
E11: =+C11/D11
F11: =+B11-E11
```

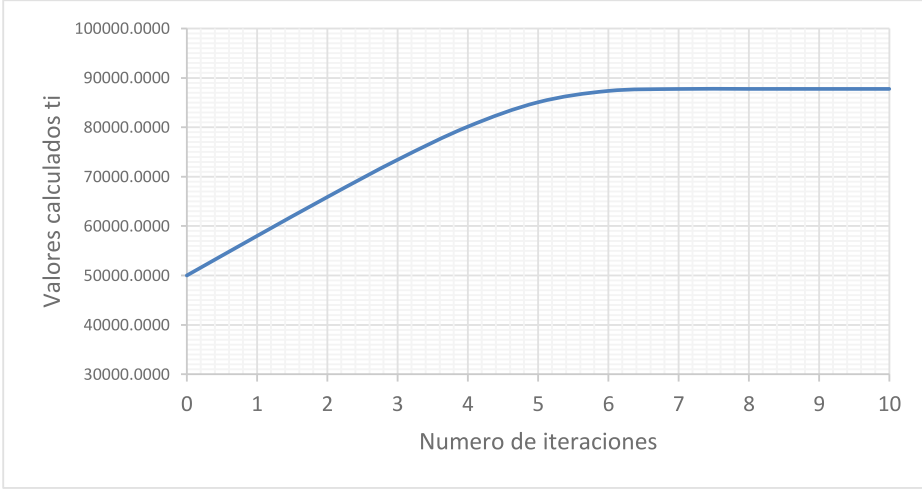


Figura 2.14.
Resultado gráfico del proceso iterativo para la ecuación de la datación paleontológica

Asimismo, establecemos un código computacional en MATLAB para la solución del algoritmo de Newton-Raphson para determinar la edad del fósil hallado **(Figura 2.15.)**.

```
clc, clear;
%Datos de entrada
l=0.00012378;          %moles/año
v=0.00000001;          %moles/año
Na=0.00010;            %moles NC14(ta)
No=1.00000;            %moles NoC14
E=0.00001;
%Proceso de cálculo e iniciación
t=50000;
d=0.01;
while abs(d)>E
F=+Na-No*(exp(1))^(-l*t)+(v/l)*(-1+(exp(1))^(-l*t));
D=l*No*(exp(1))^(-l*t)-(v/l)*(l*(exp(1))^(-l*t));
 d=F/D;
t1=t-d;
t=t1;
end
Edad_Fosil=t
```

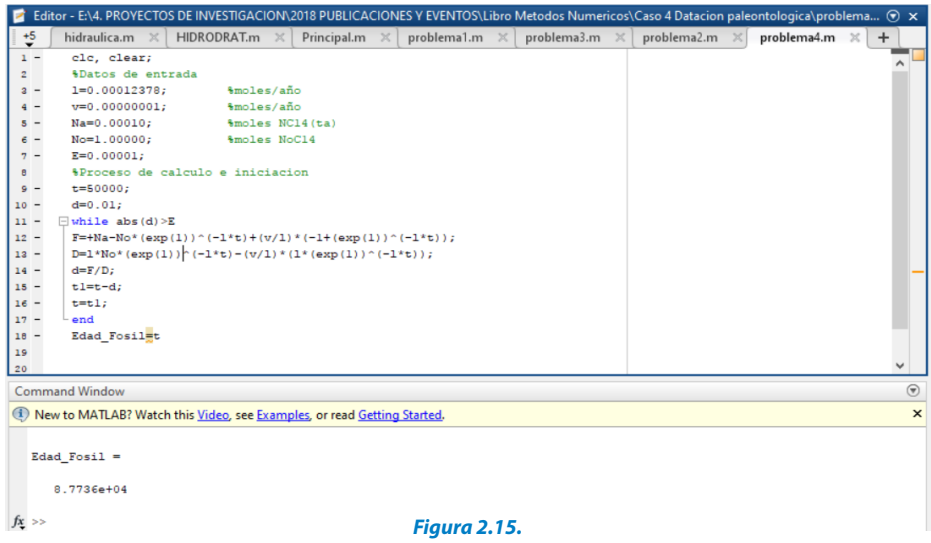


Figura 2.15.
Resultados MATLAB para la determinación de la edad del fósil

2.3.5. Estudio de Población

Formulación del caso:

Muchos campos de la ingeniería requieren estimaciones exactas de la población; por ejemplo, los ingenieros de transporte o geotécnicos de vías y pavimentos pueden considerar necesario determinar por separado la tendencia del crecimiento demográfico de una ciudad y de los suburbios adyacentes, si la población del área urbana disminuye y la suburbana crece con el tiempo, según:

$$P_u(t) = P_{u,max}e^{-k_u t} + P_{u,min} \tag{2.23}$$

$$P_s(t) = \frac{P_{s,max}}{1 + \left[P_{s,max}/P_o - 1\right]e^{-k_s t}} \tag{2.24}$$

Donde $P_{u,max}$, $P_{u,min}$, K_u, $P_{s,max}$, P_o, K_s son parámetros obtenidos en forma empírica. Determine el tiempo y los valores correspondiente a $P_u(t)$ y $P_s(t)$ cuando la población en la ciudad sea el 20 % mayor que la suburbana.

Los valores de los parámetros son:
$P_{u,max} = 100\,000$
$P\_(u,min) = 75\,000$ personas
$K_u = 0.05$/año
$P_{s,max} = 250\,000$ personas
$P_o = 5000$ personas
$K_s = 0.075$/año

Solución:

Para empezar, debemos establecer la forma de la función y su respectiva derivada:

$$F(t) = P_u(t) - 1.2\,P_s(t) = P_{u,max}e^{-k_u t} + \cdots \ldots P_{u,min} - 1.2\left[\frac{P_{s,max}}{1+\left[{}^{P_{s,max}}/_{P_o} - 1\right]e^{-k_s t}}\right] \quad (2.25)$$

$$D(t) = -k_u\,P_{u,max}e^{-k_u t} + \cdots . -1.2\left[\frac{\left[k_s\,(P_{s,max})^2 e^{-k_s t}/(P_o - 1)\right]}{\left[1+\left[{}^{P_{s,max}}/_{P_o} - 1\right]e^{-k_s t}\right]^2}\right] \quad (2.26)$$

Utilizamos la hoja de cálculo Excel para resolver el proceso iterativo de Newton-Raphson hasta lograr el valor *ti* deseado a un nivel de error preestablecido. Se muestran 10 iteraciones.

	A	B	C	D	E	F
1	**Datos de Entrada**					
2	$P_{u,max}$	100000	personas			
3	$P_{u,mim}$	75000	personas			
4	K_u	0.050	/año			
5	$P_{s,max}$	250000	personas			
6	P_o	5000	personas			
7	K_s	0.075	/año			
8	t_{i+1}	= t_i - F(t_i)/D(t_i)				
9	**Valores Iniciales**					
10	t_o=	20			Error =	0.00001
11	**Solucion Usando Newton Raphson**					
12	**i**	**t_i**	**F(t_i)**	**D(t_i)**	**F(t_i)/D(t_i)**	**t_{i+1}**
13	0	20	87114	-3537.721183738	-24.62442	44.62
14	1	44.62442	-22951	-5735.362595622	4.00159	40.62
15	2	40.62283	-738	-5346.305666185	0.13808	40.48
16	3	40.48475	-1	-5330.987297942	0.00020	40.48
17	4	40.48455	0	-5330.965245354	0.00000	40.48
18	5	40.48455	0	-5330.965245308	0.00000	40.48
19	6	40.48455	0	-5330.965245308	0.00000	40.48
20	7	40.48455	0	-5330.965245308	0.00000	40.48
21	8	40.48455	0	-5330.965245308	0.00000	40.48
22	9	40.48455	0	-5330.965245308	0.00000	40.48
23	10	40.48455	0	-5330.965245308	0.00000	40.48

Figura 2.16.
Solución Excel para la determinación del tiempo de la función combinada de la población

Con esta formulación obtenemos un tiempo de t=40.48 años. En la siguiente figura se observa el proceso iterativo hasta la convergencia hacia la solución. La formulación de las celdas C13 a F13 es:

```
C13: =+$B$2*EXP(1)^(-$B$4*B13)+$B$3-1.2*$B$5/(1+($B$5/($B$6-1))*
EXP(1)^(-$B$7*B13))
D13: =-$B$4*$B$2*EXP(1)^(-$B$4*B13)-1.2*(($B$5*$B$5/($B$6-1))*
$B$7*EXP(1)^(-$B$7*B13))/((1+($B$5/($B$6-1))*EXP(1)^
(-$B$7*B13))^2)
E13: =+C13/D13
F13: =+B13-E13
```

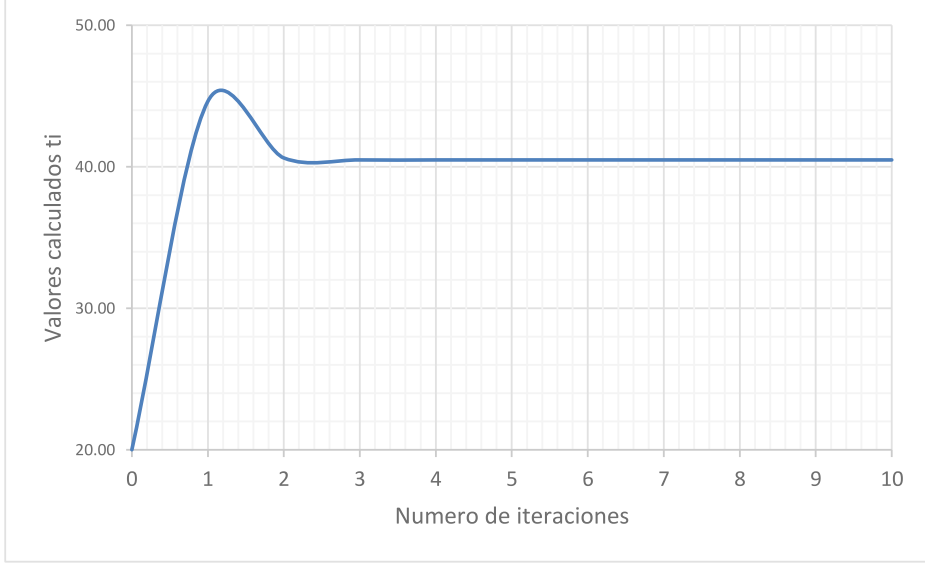


Figura 2.17.
Resultado gráfico del proceso iterativo para la ecuación combinada de la población

Asimismo, establecemos un código computacional en MATLAB para la solución del algoritmo de Newton-Raphson para determinar el tiempo en la función combinada de la población.

```
clc, clear;
%Datos de entrada
Pumax=100000;     %personas
Pumin=75000;      %personas
Ku=0.050;         %/año
Psmax=250000;     %personas
Po=5000;          %personas
Ks=0.075;         %/año
E=0.00001;
%Proceso de cálculo e iniciación
t=20;
```

```
d=0.01;
while abs(d)>E
F=Pumax*exp(1)^(-Ku*t)+Pumin-1.2*Psmax/(1+(Psmax/(Po-1))*exp(1)^(-
Ks*t));
D=-Ku*Pumax*exp(1)^(-Pumax*t)-1.2*((Psmax*Psmax/(Po-1))*Ks*exp(1)^...
            (-Ks*t))/((1+(Psmax/(Po-1))*exp(1)^(-Ks*t))^2);
d=F/D;
t1=t-d;
t=t1;
end
Put=Pumax*exp(1)^(-Ku*t)+Pumin;Pst=Psmax/(1+(Psmax/(Po-1))*exp(1)^(-
Ks*t));
Tiempo_anos=t,Poblacion_area_urbana=Put,
Poblacion_area_sub_urbana=Pst
```

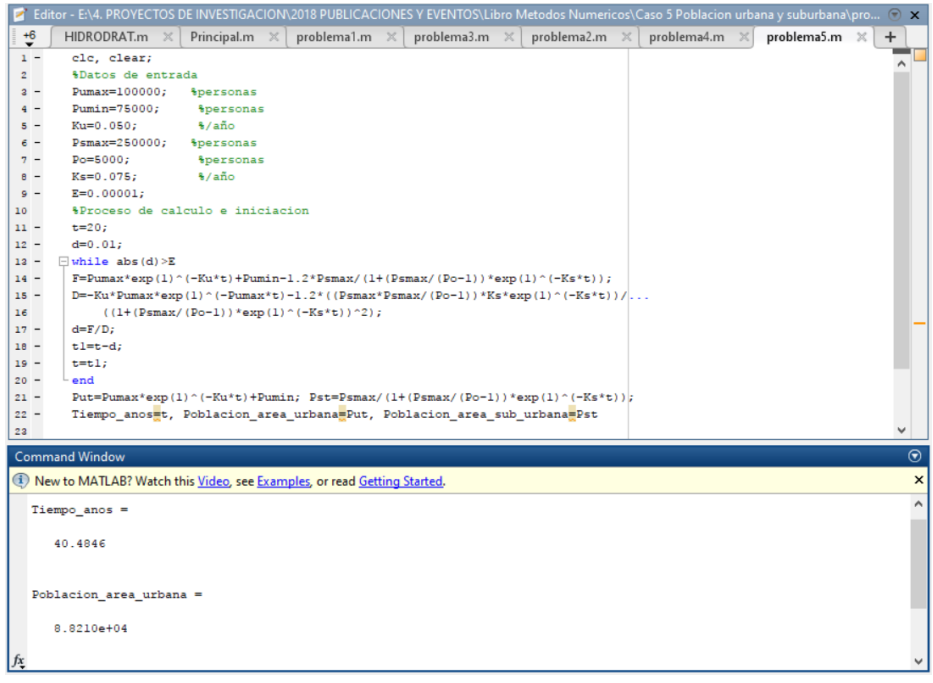


Figura 2.18.
Resultados MATLAB para la ecuación combinada de la población

2.3.6. Vertedero de Francis

Planteamiento del caso

En la hidráulica de canales abiertos, el empleo de vertederos es muy frecuente. La siguiente fórmula es atribuida a Francis y se aplica a un vertedor con contracciones.

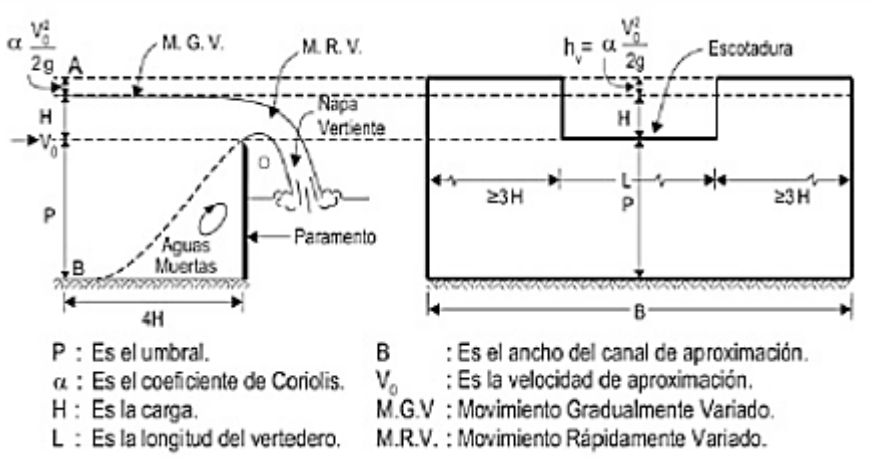


Figura 2.19.
Elementos característicos de un vertedero rectangular en pared delgada

$$Q = 3{,}33(B - 0{,}2H)(H^3)^{1/2} \tag{2.27}$$

Donde:

Q : Cantidad de agua que pasa por el vertedero en pie3/s.
B : Ancho del vertedero en pies.
H : Carga sobre la cresta del vertedero en pies.

Si se sabe que el ancho del vertedero B varía en función del caudal Q, calcule los valores de H correspondientes a las parejas de valores B y Q con el método de Newton-Raphson: (B,Q) = (3,12), (2,20), (4,13) y (5,30).

Solución:
En principio, debemos establecer la forma de la función y su respectiva derivada:

$$F(H) = 3{,}33\,B\,H^{1.5} - 0.666\,H^{2.5} - Q \tag{2.28}$$

$$D(H) = 4.995\,B\,H^{0.5} - 1.665\,H^{1.5} \tag{2.29}$$

A continuación, elaboramos una formulación en hoja de cálculo Excel que permite resolver el proceso iterativo de Newton-Raphson hasta lograr el valor *Hi* deseado a un nivel de error preestablecido.

	A	B	C	D	E	F
1	**Datos de Entrada**					
2	B	3				
3	Q	12				
4	H_{i+1}	= H_i - $F(H_i)/D(H_i)$				
5	**Valores Iniciales**					
6	H_o=	1.00			Error =	0.00001
7	**Solucion Usando Newton Raphson**					
8	**i**	**H_i**	**$F(H_i)$**	**$D(H_i)$**	**$F(H_i)/D(H_i)$**	**H_{i+1}**
9	0	1.00000	-2.67600	13.32000	-0.20090	1.20090
10	1	1.20090	0.09444	14.23024	0.00664	1.19426
11	2	1.19426	0.00009	14.20294	0.00001	1.19426
12	3	1.19426	0.00000	14.20292	0.00000	1.19426
13	4	1.19426	0.00000	14.20292	0.00000	1.19426
14	5	1.19426	0.00000	14.20292	0.00000	1.19426

Figura 2.20.
Solución Excel para la determinación del valor H del vertedero Francis

La formulación de las celdas C9 a F9 es:

```
C9: =3.33*($B$2*B9^1.5)-0.666*(B9^2.5)-$B$3
D9: =4.995*($B$2*B9^0.5)-1.665*B9^1.5
E9: =+C9/D9
F9: =+B9-E9
```

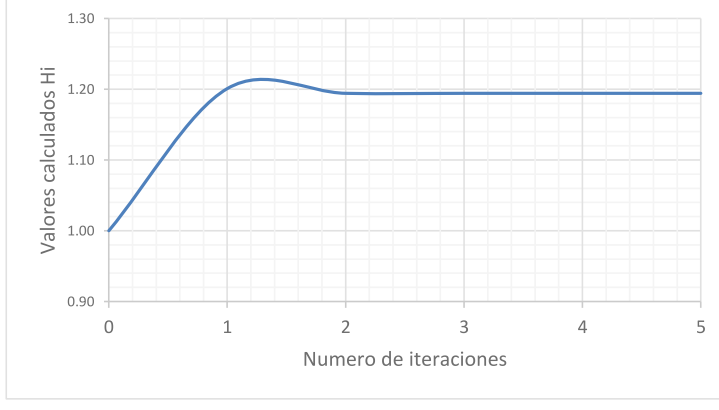


Figura 2.21.
Resultado gráfico del proceso iterativo para el valor H del vertedero Francis

Para los demás pares tenemos los siguientes resultados:

B	3	2	4	5
Q	12	20	13	30
H	1.1943	2.5278	1.0188	1.5451

Asimismo, establecemos un código computacional en MATLAB para la solución del algoritmo para determinar H.

```
clc, clear;
%Datos de entrada
B=3;
Q=12;
E=0.00001;
%Proceso de calculo e iniciacion
H=1;
d=0.01;
while abs(d)>E
F=3.33*(B*H^1.5)-0.666*(H^2.5)-Q;
D=4.995*(B*H^0.5)-1.665*H^1.5;
d=F/D;
H1=H-d;
H=H1;
end
Tirante_H=H
```

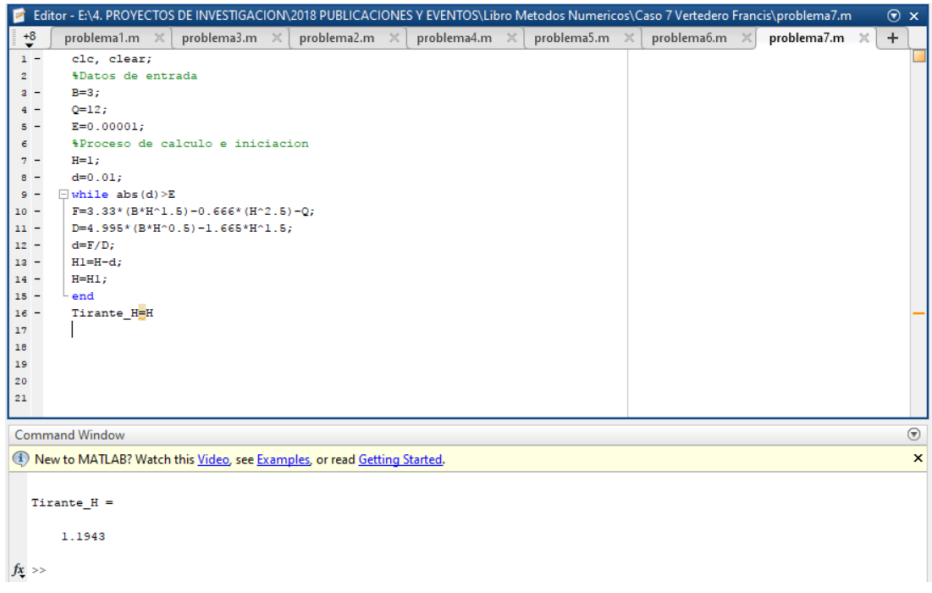


Figura 2.22.
Resultados MATLAB para el tirante H del vertedero Francis

2.3.7. Flujo Uniforme en Canales

Formulación del caso:

La ecuación mostrada representa el flujo uniforme a lámina libre, que es conocida como ecuación de Manning, en virtud a su creador:

$$Q = \frac{1}{n} A R^{2/3} S^{1/2} \tag{2.30}$$

Donde Q es el caudal en m^3/s, n el coeficiente de rugosidad de Manning adimensional. A el área hidráulica en m^2. R es el radio hidráulico ($R=A/P$) y S la pendiente longitudinal de fondo del canal.

Usar el método de Newton-Raphson para determinar el tirante de agua para una sección trapezoidal. Si se sabe que el caudal es Q=2.3 m^3/s, ancho de solera b=1.5 m, talud z=1.5, rugosidad n=0.014 y la pendiente S=0.0005.

Solución:

En primer lugar, debemos establecer la forma de la función y su derivada. Por lo tanto, la función a tratar será:

$$F = AR^{2/3} - \frac{Qn}{S^{\frac{1}{2}}} = A^3R^2 - \left(\frac{Qn}{S^{1/2}}\right)^3 = \frac{A^5}{P^2} - \left(\frac{Qn}{S^{1/2}}\right)^3 \tag{2.31}$$

$$D = \frac{A^4}{P^3}(5PT - 4AL) \tag{2.32}$$

Donde para una sección trapezoidal se cumple: $T=b+2*Z*h$, $L=(1+Z^2)^{1/2}$, $A=b*h+Z*h^2$ y $P=b+2*h*L$

A continuación, formulamos una hoja de cálculo Excel para resolver el proceso iterativo hasta lograr el valor "h" deseado. La siguiente hoja se muestra haciendo referencia al contenido de las celdas; es decir, se pueden observar las fórmulas ingresadas para lograr el valor requerido.

	A	B	C	D	E	F	G	H	I
1	**Datos de Entrada**								
2	Q=	2.3		L=	1.80277564				
3	b=	1.5		K=	2.9861568				
4	Z=	1.5							
5	S=	0.0005							
6	n=	0.014							
7	**Funcion y Derivada**								
8	F=	$\frac{A^5}{P^2} - \left(\frac{Qn}{S^{1/2}}\right)^3$							
9									
10	D=	$\frac{A^4}{P^3}(5PT - 4AL)$							
11									
12	hi+1	= hi - F(hi)/D(hi)							
13	**Valores Iniciales**								
14	ho	= b/2	0.75000	Error =	0.00001				
15	**Solucion Usando Newton Raphson**								
16	**i**	**hi**	**A (hi)**	**P (hi)**	**T (hi)**	**F(hi)**	**D(hi)**	**F(hi)/D(hi)**	**hi+1**
17	0	0.75000	1.9688	4.2042	3.7500	-1.31278	13.06670	-0.10047	0.85047
18	1	0.85047	2.3606	4.5664	4.0514	0.52948	24.61640	0.02151	0.82896
19	2	0.82896	2.2742	4.4889	3.9869	0.03288	21.61334	0.00152	0.82744
20	3	0.82744	2.2681	4.4834	3.9823	0.00015	21.41308	0.00001	0.82743
21	4	0.82743	2.2681	4.4833	3.9823	0.00000	21.41214	0.00000	0.82743
22	5	0.82743	2.2681	4.4833	3.9823	0.00000	21.41214	0.00000	0.82743

Figura 2.23.
Solución Excel para la determinación del valor H de Manning

La formulación de las celdas C17 a I17 es:

```
C17: =+$B$3*B17+$B$4*B17^2
D17: =+$B$3+2*B17*$E$2
E17: =+$B$3+2*$B$4*B17
F17: =+(C17^5/D17^2)-$E$3
G17: =+(C17^4/D17^3)*(5*D17*E17-4*C17*$E$2)
H17: =+C17/D17
I17: =+B17-E17
```

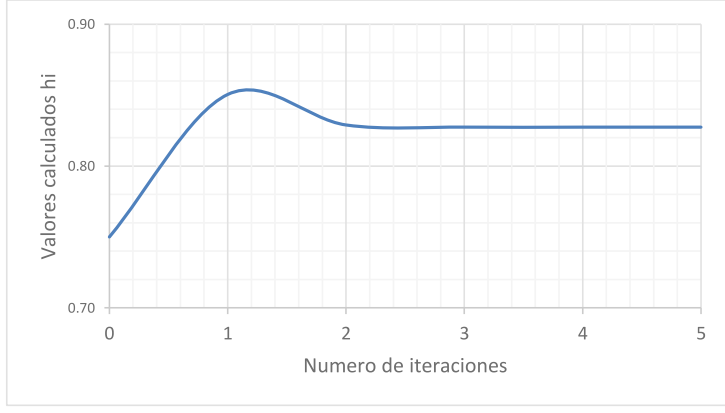


Figura 2.24.
Resultado gráfico del proceso iterativo para el valor H de Manning

Finalmente, podemos concluir que el valor del tirante de agua será de 0.8274 m. Ahora plantearemos una solución usando un código computacional en MATLAB.

```
%Datos de entrada
Q=2.3;
b=1.5;
z=1.5;
n=0.014;
S=0.0005;
%Proceso de calculo e iniciacion
h=b/2;
L=sqrt(1+z^2);
K=(Q*n/(S^0.5))^3;
E=0.00001;
d=0.01;
while abs(d)>E
A=b*h+z*h^2;
P=b+2*h*L;
T=b+2*z*h;
R=A/P;
f=(A^5/P^2)-K;
df=(A^4/P^3)*(5*P*T-4*A*L);
d=f/df;
```

```
h1=h-d;
h=h1;
end
Tirante_h=h
```

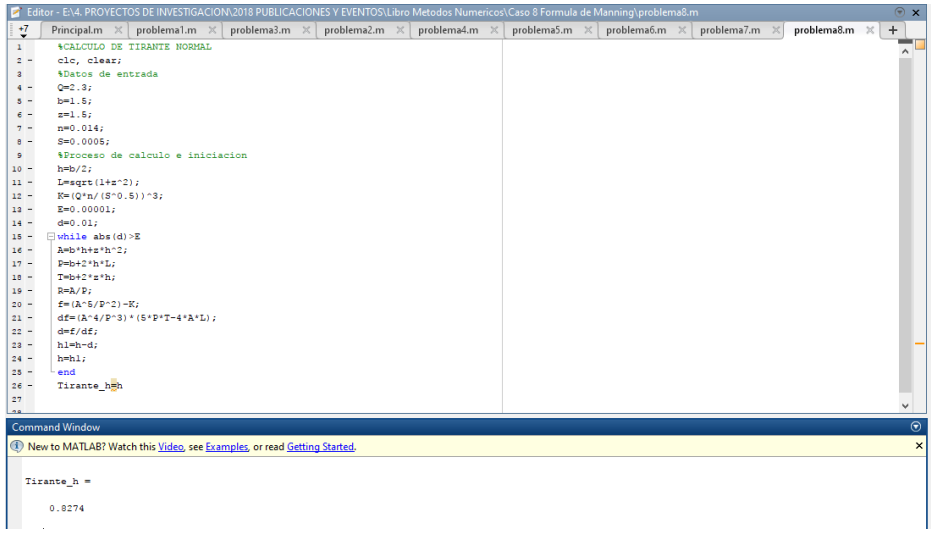


Figura 2.25.
Resultados MATLAB para el tirante H de Manning

Este programa codificado en MATLAB arroja como resultado: h = 8.2743e-001=0.8274 m. Como ejercicio de uso de las herramientas de MATLAB proponemos la siguiente solución:

```
f=inline('((1.5*h+1.5*h^2)^5)/((1.5+2*h*sqrt(1+1.5^2))^2)
-(2.3*0.014/sqrt(0.0005))^3');
y=fzero(f,1)
```

Este programa codificado en MATLAB arroja como resultado: y = 8.2743e-001=0.8274 m, similar al obtenido con el algoritmo Newton-Raphson desarrollado en MATLAB.

2.3.8. Deflexión de Vigas

Formulación del caso:

La ecuación mostrada describe el comportamiento de una viga uniforme sometida a una carga distribuida linealmente creciente. La ecuación para calcular la curva resultante es:

$$y = \frac{w_o}{120EIL}(-x^5 + 2L^2x^3 - L^4x) \tag{2.33}$$

Usar el método de Newton-Raphson para determinar el punto de máxima deflexión; es decir, el valor de x donde dy/dx=0. Usar L=450 cm, E=50 000 kN/cm^2, I=30 000 cm^4, w_0=1.75 kN/cm.

Solución:

En primer lugar, para determinar la máxima deflexión debemos plantear que dy/dx=0, esto garantizaría que obtengamos un máximo, de tal forma que esa primera derivada represente la función F y su derivada la función D.

$$F=\frac{dy}{dx}=\frac{d}{dx}\left(\frac{w_o}{120EIL}(-x^5+2L^2x^3-L^4x)\right)=\frac{w_o}{120EIL}(-5x^4+6L^2x^2-L^4) \qquad (2.34)$$

$$D=\frac{dF}{dx}=\frac{d}{dx}\left(\frac{w_o}{120EIL}(-5x^4+6L^2x^2-L^4)\right)=\frac{w_o}{120EIL}(-20x^3+12L^2x) \qquad (2.35)$$

La siguiente hoja se muestra haciendo referencia al contenido de las celdas; es decir, se pueden observar las fórmulas ingresadas para lograr luego de 5 iteraciones el valor requerido.

	A	B	C	D	E	F
1	**Datos de Entrada**					
2	L=	450	cm			
3	E=	50000	kN/cm^2			
4	I=	30000	cm^4			
5	wo=	1.75	kN/cm			
6	K	2.16049E-14	wo/(120*E*I*L)			
7	**Funcion y Derivada**					
8	F=	$\frac{w_o}{120EIL}(-5x^4+6L^2x^2-L^4)$				
9						
10	D=	$\frac{w_o}{120EIL}(-20x^3+12L^2x)$				
11						
12	Xi+1	= Xi - F(Xi)/D(Xi)				
13	**Valores Iniciales**					
14	Xo	= L/2 =	225.0		Error = 0.00001	
15	**Solucion Usando Newton Raphson**					
16	**i**	**Xi**	**F(Xi)**	**D(Xi)**	**F(Xi)/D(Xi)**	**Xi+1**
17	0	225.00000	0.000166	0.000007	24.107143	200.89286
18	1	200.89286	-0.000002	0.000007	-0.353262	201.24612
19	2	201.24612	0.000000	0.000007	0.000001	201.24612
20	3	201.24612	0.000000	0.000007	0.000000	201.24612
21	4	201.24612	0.000000	0.000007	0.000000	201.24612
22						
23						
24	CALCULO DE Y_{MAX}		-0.11411	cm	$y=\frac{w_o}{120EIL}(-x^5+2L^2x^3-L^4x)$	

Figura 2.26.
Solución Excel para la determinación del valor de la posición de la flecha máxima y el valor de esta.

La formulación de las celdas C17 a F17 es:

```
C17: =(+$B$6*(-5*B17^4+6*$B$2^2*B17^2-$B$2^4))
D17: =+$B$6*(-20*B17^3+12*$B$2^2*B17)
E17: =+C17/D17
F17: =+B17-E17
```

De esta forma podemos concluir que el valor de la máxima deflexión en la viga será de 0.1141 cm hacia abajo.

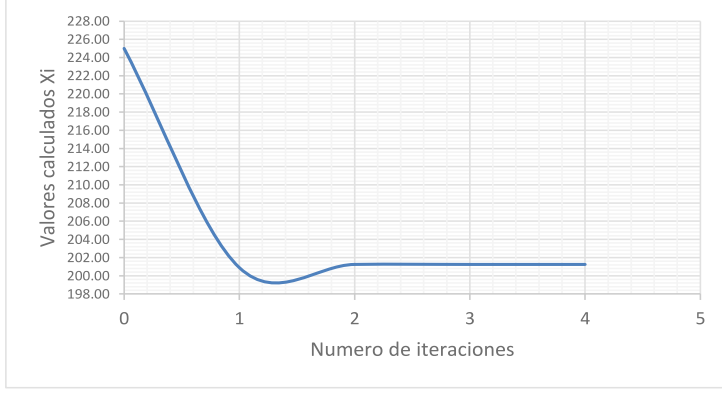


Figura 2.27.
Resultado gráfico del proceso iterativo para el valor de la posición de la flecha máxima.

Ahora plantearemos una solución estableciendo un código computacional en MATLAB para la solución del algoritmo.

```
clc, clear; %Datos de entrada
L=450;
E=50000;
I=30000;
wo=1.75;
K=wo/(120*E*I*L);
E=0.0001;
%Proceso de cálculo e iniciación
x=L/2; %valor inicial
d=0.01;
while abs(d)>E
F=K*(-5*x^4+6*L^2*x^2-L^4);
D=K*(-20*x^3+12*L^2*x);
d=F/D;
x1=x-d;
x=x1;
end
x
Df=K*(-x^5+2*L^2*x^3-L^4*x) %calcula la deflexión máxima en x
calculado
```

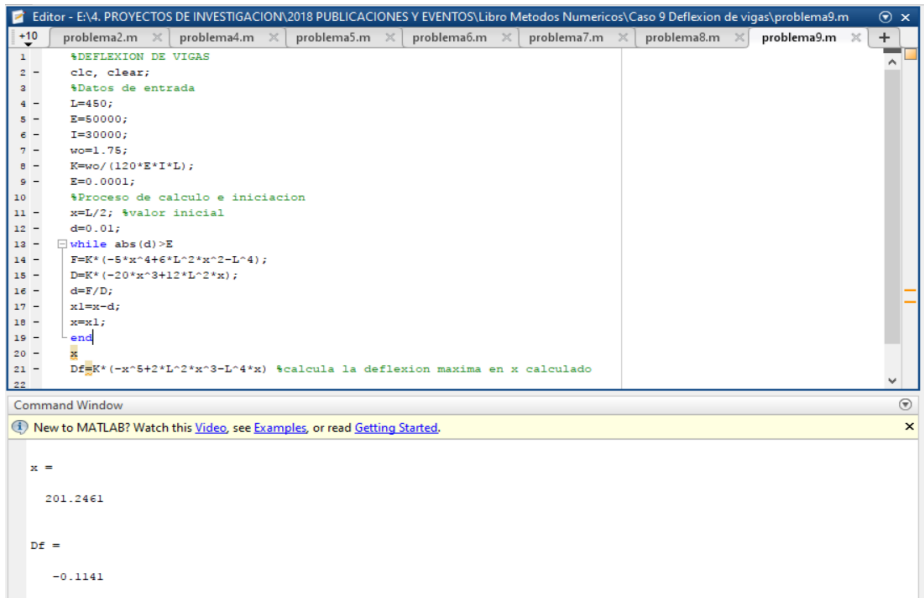


Figura 2.28.
Resultados MATLAB para la posición de la flecha máxima y el valor de esta.

Este programa codificado en MATLAB arroja como resultado: x = 2.0125e+002, Df = -1.1411e-001. Por otro lado, usando funciones de MATLAB:

```
clc, clear
f=inline('-5*x^4+6*450^2*x^2-450^4');
x=fzero(f,150)
```

Obtenemos como resultado: x = 2.0125e+002=201.25 cm, lo que resulta un procedimiento rápido y eficiente.

2.3.9. Vibración Amortiguada

Formulación del caso:

El desplazamiento de una estructura está definido por la siguiente ecuación para una vibración amortiguada:

$$f(t) = 8e^{-kt}Cos\,(wt) \tag{2.36}$$

Usar el método de Newton-Raphson para determinar el tiempo transcurrido para que el desplazamiento disminuya a 4 con k=0.5 y w=3.

Solución:

Para la solución requerimos la función F y la derivada D de esta función.

$$F = 8e^{-kt}Cos\,(wt) - 4 \tag{2.37}$$

$$D = e^{-kt}[-24\,Sen(wt) - 4\,Cos\,(wt)] \tag{2.38}$$

La siguiente hoja Excel muestra el desarrollo del algoritmo de Newton-Raphson y un total de 6 iteraciones para conseguir la solución buscada.

	A	B	C	D	E	F
1	**Datos de Entrada**					
2	w	3				
3	k	0.5				
4	t_{i+1}	= t_i - F(t_i)/D(t_i)				
5	**Valores Iniciales**					
6	t_o=	0.50			Error =	0.00001
7	**Solucion Usando Newton Raphson**					
8	**i**	**t_i**	**F(t_i)**	**D(t_i)**	**F(t_i)/D(t_i)**	**t_{i+1}**
9	0	0.50000	-3.55928	-18.86476	0.18867	0.31133
10	1	0.31133	0.07136	-18.54996	-0.00385	0.31517
11	2	0.31517	-0.00014	-18.62207	0.00001	0.31517
12	3	0.31517	0.00000	-18.62193	0.00000	0.31517
13	4	0.31517	0.00000	-18.62193	0.00000	0.31517
14	5	0.31517	0.00000	-18.62193	0.00000	0.31517

Figura 2.29.
Solución Excel para la determinación del tiempo en el que se alcanza un desplazamiento de 4 unidades.

La formulación de las celdas C9 a F9 es:

```
C9: =(8*((EXP(1))^(-$B$3*B9))*COS($B$2*B9))-4
D9:  =((EXP(1))^(-$B$3*B9))*(-24*SENO($B$2*B9)-
4*COS($B$2*B9))
E9: =+C9/D9
F9: =+B9-E9
```

De esta forma podemos concluir que el tiempo para obtener un desplazamiento de la estructura de 4 unidades será de 0.31517 segundos.

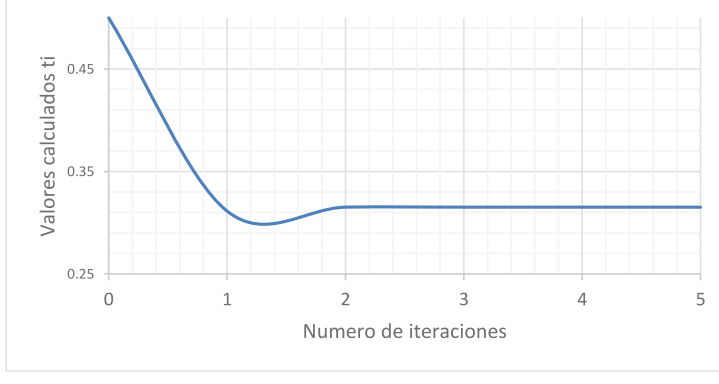


Figura 2.30.
Resultado gráfico del proceso iterativo para el valor del tiempo de desplazamiento.

A continuación, se muestra un código computacional en MATLAB para la solución del algoritmo.

```
clc, clear;
%Datos de entrada
w=3;
k=0.5;
E=0.00001;
%Proceso de cálculo e iniciación
t=k;
d=0.01;
while abs(d)>E
F=(8*((exp(1))^(-k*t))*cos(w*t))-4;
D=((exp(1))^(-k*t))*(-24*sin(w*t)-4*cos(w*t));
d=F/D;
t1=t-d;
t=t1;
end
Tiempo_Desplazamiento=t
```

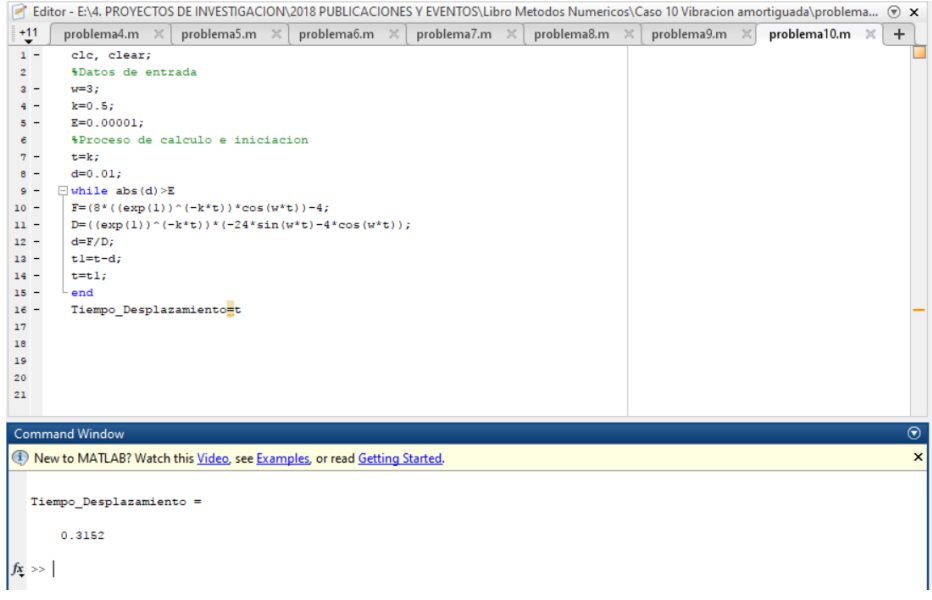


Figura 2.31.
Resultados MATLAB para el tiempo de desplazamiento establecido.

Este programa codificado en MATLAB arroja como resultado: $t = 0.3152$.

2.4. Problemas Propuestos

- **Problema propuesto 1:** La energía libre de Gibb´s para una molécula de hidrógeno a temperatura T es:

$$G = -RT\ln\left[(T/T_0)^{5/2}\right] \text{ J}$$

Donde R = 8.31441, J/K es la constante del gas y T_0=4.44418 °K. Determine la temperatura T para G=-$10^5 J$

- **Problema propuesto 2:** La velocidad de caída de un paracaidista está dada por:

$$v = \frac{g \cdot m}{c}\left(1 - e^{-(c/m)\cdot t}\right)$$

Donde g=9.8 m/s^2, para el paracaidista con un coeficiente de arrastre c= 14 kg/s, calcule la masa (m) de este de tal forma que la velocidad sea de 35 m/s en t=7 s. Usar método Newton-Raphson con un nivel de error Es=0.1 %. Es posible que la masa se encuentre entre 50 y 100 Kg, utilizar estos valores como referencia para el valor inicial.

- **Problema propuesto 3:** En la hidráulica de canales es frecuente calcular el tirante crítico del flujo con fines de análisis y diseño. Si se sabe que el régimen crítico en canales trapezoidales está gobernado por las siguientes ecuaciones:

Función: $f(y) = \frac{A^3}{T} - \frac{Q^2}{g}$ Derivada: $f'(y) = 3 \cdot A^2 - \frac{2 \cdot z \cdot A^3}{T^2}$

Área: $A = (b + z \cdot y)y$; Espejo: $T = b + 2 \cdot z \cdot y$

Utilizando el Algoritmo de Newton-Raphson determinar el tirante crítico "y" que se encuentra implícito en las ecuaciones del régimen crítico. Para el cálculo iniciar con y=1.0 m y utilizar los siguientes datos: caudal Q= 150.4 m^3/s, ancho de base b= 7.4 m y talud z= 3, donde Área (A) y Espejo (T) están en función de: "y", "b" y "z".

- **Problema propuesto 4:** El factor de fricción f de Darcy es utilizado para obtener la pérdida de carga en tuberías. Determinar este factor f utilizando la ecuación implícita de Colebroock-White con una aproximación de $|e_r| \leq 0{,}001\%$. Para el cálculo de f utilizar los siguientes datos: caudal de agua Q= 0.042 m^3/s, rugosidad de tubería de PVC Ks=1.5x10^{-6} m, diámetro d= 0.1524 m y viscosidad cinemática del agua v=1.14x10^{-6} m^2/s.

La ecuación de Colebroock-White es:

$$\frac{1}{\sqrt{f}} = -2\log_{10}\left(\frac{Ks}{3{,}71\cdot d} + \frac{2{,}51}{\mathrm{Re}\cdot\sqrt{f}}\right)$$

Adicionalmente Re = Número de Reynolds (adimensional y constante) $\mathrm{Re} = \frac{V\cdot d}{v}$.

La ecuación de continuidad: $Q = A\cdot V$ donde A = área y V = velocidad del flujo en la tubería. Fórmulas adicionales de la función y derivada de la fórmula de Colebroock-White:

$$x = \frac{1}{\sqrt{f}}$$

Si hacemos:

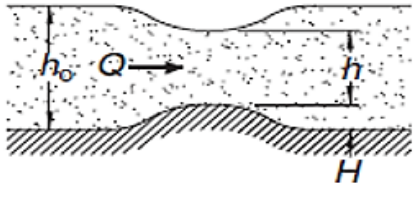
$$g(x) = -2\log_{10}\left(\frac{Ks}{3{,}71d} + \frac{2{,}51x}{\mathrm{Re}}\right) - x$$

$$g'(x) = -\frac{2}{\ln 10}\left(\frac{\frac{2{,}51}{\mathrm{Re}}}{\frac{Ks}{3{,}71d} + \frac{2{,}51x}{\mathrm{Re}}}\right) - 1$$

Algoritmo de Newton-Raphson:

$$x_{i+1} = x_i - \frac{g(x_i) - x_i}{g'(x_i) - 1}$$

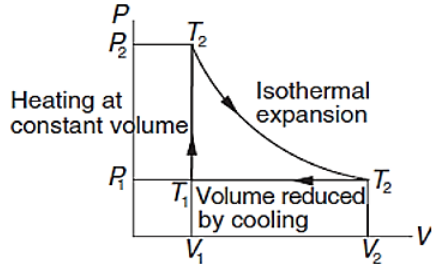
- **Problema propuesto 5:** La ecuación de Bernoulli para el flujo de fluidos en un canal abierto con una pequeña protuberancia dada de la siguiente manera:



$$\frac{Q^2}{2gb^2h_0^2} + h_0 = \frac{Q^2}{2gb^2h^2} + h + H$$

Dónde Q=1.2 m^3/s, g=9.81 m/s^2, b=1.8 m, h_0=0.6 m, H=0.075 m y h=tirante aguas abajo de la protuberancia. Usando Newton-Raphson, calcular "h".

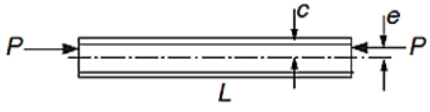
- **Problema propuesto 6:** La figura muestra el ciclo termodinámico de un motor. La eficiencia de este motor para un gas es:



$$\eta = \frac{\ln(T_2/T_1) - (1 - T_1/T_2)}{\ln(T_2/T_1) + (1 - T_1/T_2)/(\gamma - 1)}$$

Donde T es la temperatura absoluta y $y = 5/3$. Encuentra la relación T_2/T_1 que dé como resultado un 30 % de eficiencia.

- **Problema propuesto 7:** Una barra de aluminio W310x202 (brida ancha) está sujeta a una carga axial excéntrica P como se muestra en la figura. La tensión de compresión máxima en la columna viene dada por la siguiente fórmula:



$$\sigma_{\max} = \bar{\sigma}\left[1 + \frac{ec}{r^2}\sec\left(\frac{L}{2r}\sqrt{\frac{\bar{\sigma}}{E}}\right)\right]$$

Donde:
σ = P/A = Esfuerzo promedio.
A = 25 800 mm^2 = Área de la sección transversal.
e = 85 mm = Excentricidad de la carga.
c = 170 mm = Profundidad media de la barra.
r = 142 mm = Radio de giro de la sección transversal.
L = 7100 mm = Longitud de la barra.
E = 71 x 10^9 Pa = Módulo de Elasticidad.

Determine la máxima carga P que la barra puede transportar si el esfuerzo máximo no debe exceder 120 x 10^6 Pa.

CAPÍTULO III
ECUACIONES DIFERENCIALES ORDINARIAS (EDO)

3.1. Generalidades

Las leyes fundamentales de la física, mecánica, electricidad, termodinámica y otros están basadas con frecuencia en observaciones experimentales que explican variaciones en las propiedades físicas y estados de los sistemas. Estas leyes, más que describir directamente el estado de los sistemas físicos, se expresan en términos de los cambios espaciales y temporales de las variables intervinientes (Chapra & Canale, 2005).

Es por ello que las ecuaciones diferenciales tienen importancia fundamental en las aplicaciones de las ciencias e ingeniería, ya que numerosos procesos físicos son idealizados (o modelizados) matemáticamente por estas ecuaciones. Tales ecuaciones son a veces conocidas como ecuaciones de razón, ya que expresan la razón de cambio de una variable como una función de las variables y parámetros del problema.

Podemos citar como ejemplos de las leyes fundamentales que se escriben en términos de la razón de cambio de las variables:

- Segunda ley de Newton del movimiento.

$$\frac{dv}{dt} = \frac{F}{m} \tag{3.39}$$

Donde v es la velocidad, F es la fuerza, m es la masa y t el tiempo.

- Ley del calor de Fourier.

$$q = -k' \frac{dT}{dx} \tag{3.40}$$

Donde q es el flujo de calor, k' es la conductividad térmica, T es la temperatura y x la variable espacial.

- Ley de difusión de Fick.

$$J = -D \frac{dc}{dx} \tag{3.41}$$

Donde J es el flujo másico, D es el coeficiente de difusión, c es la concentración y x la variable espacial.

- Ley de Faraday.

$$\delta V_L = L \frac{di}{dt} \tag{3.42}$$

Donde δV_L es la caída de voltaje, L es la inductancia, i es la corriente y t la variable temporal.

La mayoría de las ecuaciones diferenciales de importancia práctica no se pueden resolver mediante métodos analíticos de cálculo, y es debido a esto que los métodos numéricos han adquirido una importancia extraordinaria en todos los campos de las ciencias e ingeniería, sobre todo a partir de la disponibilidad de computadoras que soportan grandes volúmenes de cálculo.

3.2. Uso de las EDO en las Ciencias e Ingeniería

En la solución de muchos problemas en las ciencias e ingeniería es común encontrarnos con ecuaciones diferenciales ordinarias, para las cuales existen diversas técnicas que permiten hallar la solución en términos de funciones elementales o especiales (Chapra & Canale, 2005).

Al intentarse la solución de estas ecuaciones, en algunas ocasiones, no es posible resolverlas por medio de los métodos clásicos por ser la solución muy difícil de obtener o tan difícil y laboriosa en su desarrollo que no justifican el esfuerzo empleado en la resolución. En estos casos la solución numérica de las EDO queda plenamente justificada. Por otro lado, la solución numérica de las ecuaciones diferenciales no dispensa la responsabilidad de formular correctamente el problema, ni será la excusa para un análisis del mismo mal formulado.

3.3. Definiciones de las EDO

Solo a manera de síntesis de lo estudiado en los cursos de matemática superior, se enumera, a continuación, un conjunto de definiciones acerca de las ecuaciones diferenciales, su caracterización y solución (Plaat, 1974).

- Ecuaciones diferenciales ordinarias y parciales: Si en una ecuación diferencial hay una sola variable independiente, las derivadas son totales y la ecuación se denomina *ordinaria*. Por el contrario, si aparecen dos o más variables independientes, las derivadas serán *parciales* y la ecuación será *diferencial parcial*.
- Orden de una ecuación diferencial: Es la derivada de mayor orden que aparece en la ecuación.
- Ecuación diferencial lineal: Una ecuación diferencial es lineal si en ella no aparecen potencias de la variable dependiente ni de sus derivadas ni productos de la variable dependiente por sus derivadas o productos entre derivadas. Una ecuación diferencial ordinaria lineal es aquella que se ajusta a la forma general:

$$a_n(x)\cdot y^{(n)} + \ldots\ldots + a_1(x)\cdot y' + a_0(x)\cdot y = f(x) \qquad (3.43)$$

Donde "$y(n)$" es la n-ésima derivada de y con respecto "ax", y "$a(x)$" y "$f(x)$" son funciones específicas de "x".

- Solución de una ecuación diferencial: Es cualquier relación funcional que no incluya derivadas o integrales de funciones desconocidas y que la verifique idénticamente por sustitución directa.

3.4. Solución de una EDO

Para exponer el método de solución de una EDO, ya sea de primer orden u orden superior, disponemos de varios métodos, desde los clásicos métodos de Euler y Heun hasta los métodos multipasos más avanzados (Jorquera & Weston, 2014).

De esta manera presentaremos los métodos propuestos a la fecha, invocando a la solución de problemas de ciencias e ingeniería, utilizando herramientas como la hoja de cálculo Excel y la programación en MATLAB.

Para exponer los métodos de solución numérica de estas ecuaciones consideraremos una EDO de primer orden de la forma:

$$y' = \frac{dy}{dx} = f(x, y) \tag{3.44}$$

De la cual es fundamental expresar que se conoce una condición inicial:

$$y(x_0) = y_0 \tag{3.45}$$

En los casos en que se disponga de ecuaciones de segundo orden, estas pueden reducirse a un sistema de ecuaciones simultáneas de primer orden del siguiente modo:

Consideremos una EDO de segundo orden de la forma:

$$y'' = \frac{d^2y}{dx^2} = f(y', y, x) \tag{3.46}$$

En la cual utilizando un adecuado cambio de variable podemos establecer:

$$z' = f(z, y, x) \tag{3.47}$$

La cual, evidentemente, ha sido transformada y representa una EDO de primer orden.

3.5. Convergencia de una EDO

Es habitual que en los métodos numéricos se busque garantizar la convergencia; esta se alcanza con consistencia y estabilidad: Consistencia + Estabilidad = Convergencia. Sin embargo, tanto al interpretar el concepto de estabilidad como el

de convergencia, y hacer uso de este resultado, lo mismo que ocurre en el marco de la solución de las ecuaciones diferenciales, nos enfrentamos a genuinos problemas de dimensión infinita, y la elección que se hace de los parámetros es fundamental. Así, estos tres conceptos han de ser manipulados en un mismo contexto una vez establecidas con claridad las normas en las que trabajamos, lo cual, en realidad, consiste en determinar el criterio o distancia en la que se va a comprobar la convergencia del método.

La consistencia del método numérico hace referencia a su coherencia a la hora de aproximar la ecuación diferencial (ED). Se trata simplemente de comprobar si el esquema numérico utilizado es un esquema razonable para aproximar la ED en cuestión o, si, por el contrario, corre el riesgo de aproximar a otra ED.

La estabilidad, por sí sola, no basta para garantizar la convergencia del método. Es preciso analizar su estabilidad. La propiedad de estabilidad consiste en asegurarse que los esquemas discretos o semidiscretos, en su evolución temporal (discreta o continua), no amplifiquen los errores iniciales o, al menos, no lo hagan de manera creciente y descontrolada a medida que el tamaño del paso tiende a cero.

3.6. Métodos de Solución Numérica de las EDO

A continuación, listaremos los métodos de solución para las EDO.

3.6.1. Método de Euler ("Predictor")

En este primer método expondremos la deduccion del método en forma detallada, para lo cual consideramos una EDO de primer orden de la forma:

$$y_i' = \frac{dy_i}{dx_i} = f(x_i, y_i) \tag{3.48}$$

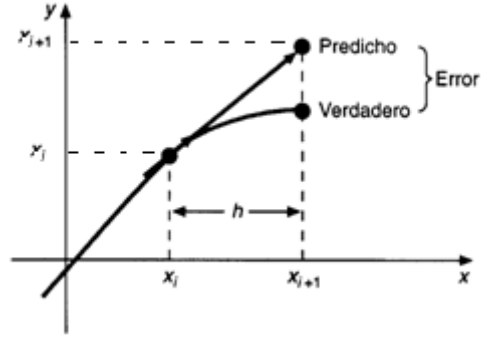


Figura 3.1.
Procedimiento del método de Euler.

Denominemos y_{i+1} al punto en que la recta tangente intercepta a la línea vertical levantada del punto $x_{i+1} = x_i + h$, por tanto, podemos plantear la ecuación de la recta tangente en el punto conocido a la curva buscada:

$$y = y_i + y_i'(x - x_i) \tag{3.49}$$

Expresión en la cual y_i' corresponde a la primera derivada de la función. Además, de acuerdo a la condición inicial se tiene que: $h=x_{i+1} - x_i$. Finalmente, reemplazando en la ecuación de la recta, tenemos:

$$y_{i+1}^* = y_i + h.f(x_i, y_i) \tag{3.50}$$

De esta manera podemos acotar que el valor de h es elegido a criterio del calculista, con las consideraciones teóricas que representa su elección.

El procedimiento a seguir para su aplicación es el siguiente:

- Primero: Consideremos conocida una ecuación diferencial de primer orden de la forma $y_i' = \frac{dy_i}{dx_i} = f(x_i, y_i)$, además de conocer un punto de la curva (x_i, y_i).
- Segundo: Designemos conveniente una cierta longitud "h" entre los valores puntuales a determinarse a lo largo del eje X.
- Tercero: Hallaremos el nuevo valor de "y", mediante la expresión: $y_{i+1}^* = y_i + h.f(x_i, y_i)$, de este modo se habrá determinado el valor del nuevo punto: (x_{i+1}, y_{i+1}), $x_{i+1} = x_i + h$ y para $y_{i+1} = y^*$.
- Cuarto: Repetimos el procedimiento hasta que el valor de x se estime conveniente.

Pseudocódigo:

$Euler(a, b, N, \alpha)$
$h \leftarrow (b - a)/N$
$t_0 \leftarrow a$
$y_0 \leftarrow \alpha$
Para i desde 1 hasta N hacer:
$x_{i+1} \leftarrow x_i + h$
$y_{i+1} \leftarrow y_i + h * f(x_i, y_i)$
Fin, parar
Mostrar $(t_0, y_0), (t_1, y_1), \ldots \ldots (t_N, y_N)$
FIN

3.6.2. Método de Heun ("Predictor-Corrector")

Como ya hemos mencionado, la fuente fundamental de error en el método de Euler es que la derivada de la función, al inicio del intervalo, se aplica a través de toda la longitud del mismo. El método de Heun propone mejorar la estimación de la pendiente calculando las derivadas al comienzo y al final del intervalo. Luego, estas derivadas se promedian para obtener una estimación mejorada de la pendiente para todo el intervalo. Este procedimiento se ilustra en la **Figura 3.2.**

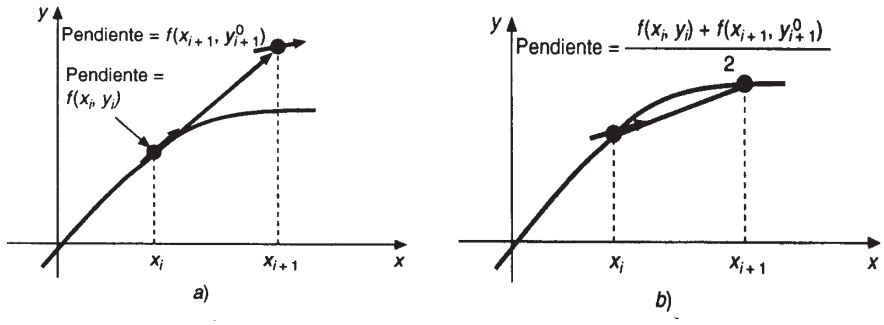


Figura 3.2.
Procedimiento del método de Heun.

Considerando un punto Q de coordenadas (x_{i+1}, y_{i+1}), respecto al de referencia, la pendiente que pasa por estos puntos será:

$$Pendiente = \frac{1}{2}[f(x_i, y_i) + f(x_i + h, y_i + hy_i')] = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \qquad (3.51)$$

Donde $h = x_{i+1} - x_i$, entonces tendremos:

$$y_{i+1} = y_i + \frac{h}{2}[f(x_i, y_i) + f(x_{i+1}, y^*_{i+1})] \qquad (3.52)$$

Pseudocódigo:

$Heun(a, b, N, \alpha)$
$h \leftarrow (b - a)/N$
$t_0 \leftarrow a$
$y_0 \leftarrow \alpha$
Para i desde 1 hasta N hacer:
$x_{i+1} \leftarrow x_i + h$
$y_{(i+1)} \leftarrow y_i + (h/2)(f(x_i, y_i) + f(x_{i+1}, y^*_{i+1}))$
Fin, parar
Mostrar $(t_0, y_0), (t_1, y_1), \ldots\ldots (t_N, y_N)$
FIN

3.6.3. Método de Runge-Kutta de 1.er Orden

El método de Heun, el del punto medio y la misma técnica de Euler son casos particulares de una clase general de procedimientos de un solo paso; todos ellos denominados métodos de Runge-Kutta. Esta afirmación será demostrada en los desarrollos subsiguientes.

Los métodos de Runge-Kutta (RK) tienen la característica de poseer precisiones propias de desarrollos de Taylor, que incluyen términos de derivadas de órdenes superiores a uno, sin requerir el cálculo de las mismas.

Si escribimos la ecuación original del método de Euler en forma generalizada, tendremos:

$$y_{i+1} = y_i + \varphi(x_i, y_i, h)h \tag{3.53}$$

Donde φ se la denomina función incremento, y puede ser interpretada como una pendiente representativa sobre el intervalo. La función incremento se escribe en general como:

$$\varphi = a_1 k_1 + a_2 k_2 + \cdots \ldots\ldots + a_n k_n \tag{3.54}$$

Donde "$a_1, a_2, \ldots.. a_n$" son constantes y las k quedan representadas por:

$$\begin{aligned} k_1 &= f(x_i, y_i) \\ k_2 &= f(x_i + p_1 . h, y_i . q_{11} k_1 . h) \\ k_3 &= f(x_i + p_2 . h, y_i + q_{21} k_1 . h + q_{22} k_2 . h) \\ k_3 &= f(x_i + p_2 . h, y_i + q_{21} k_1 . h + q_{22} k_2 . h) \end{aligned} \tag{3.55}$$

$$k_n = f(x_i + p_{n-1} . h, y_i + q_{n-1,1} k_1 . h + q_{n-1,2} k_2 . h + \cdots \ldots\ldots + q_{n-1,n-1} k_{n-1} . h)$$

Finalmente, manteniendo la nomenclatura de los métodos de Euler y Heun, el siguiente esquema para el primer método Runge-Kutta.

$$\begin{aligned} k_1 &= hf(x_o , y_o) \\ k_2 &= hf(x_o + \frac{1}{2}h , y_o + \frac{1}{2}k_1) \\ k_3 &= hf(x_o + \frac{1}{2}h , y_o + \frac{1}{4}k_1 + \frac{1}{4}k_2) \\ k_4 &= hf(x_o + h , y_o - k_2 + 2k_3) \\ y_1 &= y_o + \frac{1}{6}(k_1 + 4k_3 + k_4) \end{aligned} \tag{3.56}$$

Pseudocódigo:
Este pseudocódigo es válido para todos los métodos RK, haciendo la sustitución de ecuaciones respectiva.

1. RK Proporcionar (f, x_o, y_o, h, n)
2. Imprimir x_o, y_o
3. Desde $i = 1$ *hasta* $i = n$
a. Calcular:
$k_1 = hf(x_o , y_o)$
$k_2 = hf(x_o + \frac{1}{2}h , y_o + \frac{1}{2}k_1)$
$k_3 = hf(x_o + \frac{1}{2}h , y_o + \frac{1}{4}k_1 + \frac{1}{4}k_2)$
$k_4 = hf(x_o + h , y_o - k_2 + 2k_3)$
$y_1 = y_o + \frac{1}{6}(k_1 + 4k_3 + k_4)$
b. Hacer $y_{o=} y_1$; $x_{o=} x_o + h$;
c. Imprimir x_o, y_o
4. Terminar

3.6.4. Método de Runge-Kutta de 2.° Orden

En forma similar al primer método, establecemos el sistema de ecuaciones de solución para el segundo método:

$$\begin{aligned} k_1 &= hf(x_0, y_0) \\ k_2 &= hf\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_1\right) \\ k_3 &= hf\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2\right) \\ k_4 &= hf(x_0 + h, y_0 + k_3) \\ y_1 &= y_0 + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned} \tag{3.57}$$

3.6.5. Método de Runge-Kutta de 3.er Orden

Se puede hacer un desarrollo similar al método de segundo orden. Como resultado de dicho desarrollo se llega a 6 ecuaciones con 8 incógnitas, por lo que deben especificarse con antelación los valores de 2 de ellas con el fin de establecer todos los parámetros restantes. Una versión común del método Runge-Kutta de tercer orden resultante es:

$$\begin{aligned} k_1 &= f(x_i, y_i) \\ k_2 &= f\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}hk_1\right) \\ k_3 &= f(x_i + h, y_i - k_1h + 2k_2h) \\ y_{i+1} &= y_i + \frac{h}{6}(k_1 + 4k_2 + k_3) \end{aligned} \tag{3.58}$$

Puede observarse que, si la derivada de la función solución depende solo de x, este método de tercer orden se reduce a la regla de Simpson 1/3. Los métodos de Runge-Kutta de tercer orden tienen errores locales y globales de $O(h4)$ y $O(h3)$, respectivamente, y dan resultados exactos cuando la función es una función cúbica o de menor orden. Si se trata de polinomios, la ecuación anterior dará también resultados exactos cuando la función solución de la ecuación diferencial es de cuarto orden, debido a que la regla de Simpson 1/3 proporciona estimaciones exactas de la integral de funciones cúbicas.

3.6.6. Método de Runge-Kutta de 4.° Orden

De las versiones de los métodos de Runge-Kutta, el de cuarto orden es el más utilizado. Puede observarse que, si la derivada de la función solución depende solo de x, el método RK clásico de cuarto orden es similar a la regla de Simpson 1/3. También presenta alguna similitud con el método de Heun, en el sentido que son desarrolladas estimaciones múltiples de las pendientes en el punto medio para, finalmente, combinadas con las obtenidas al inicio y final del intervalo, obtener la pendiente promedio mejorada para el intervalo. En esta versión del método, como en las anteriores de distintos órdenes, cada una de las

k_i representa una pendiente. Luego, reemplazándolas en la primera expresión, se obtiene una pendiente media mejorada representativa del intervalo.

$$\begin{aligned} k_1 &= f(x_i, y_i) \\ k_2 &= f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}h.k_1) \\ k_3 &= f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}hk_2) \\ k_4 &= f(x_i + h, y_i + k_3 h) \\ y_{i+1} &= y_i + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned} \tag{3.59}$$

3.6.7. Método de Runge-Kutta de Orden Superior

Si se requiere mayor exactitud en las estimaciones, es recomendable utilizar alguno de los métodos de Runge-Kutta de quinto orden. Entre estos se destaca el método de Butcher. Este método es más preciso que el de cuarto orden, pero es mucho mayor la complejidad y el esfuerzo computacional.

$$\begin{aligned} k_1 &= f(x_i, y_i) \\ k_2 &= f(x_i + \frac{1}{4}h, y_i + \frac{1}{4}h.k_1) \\ k_3 &= f(x_i + \frac{1}{4}h, y_i + \frac{1}{8}hk_1 + \frac{1}{8}hk_2) \\ k_4 &= f\left(x_i + \frac{1}{2}h, y_i - \frac{1}{2}k_2 h\right) + k_3 h \\ k_5 &= f\left(x_i + \frac{3}{4}h, y_i + \frac{3}{16}k_1 h + \frac{3}{16}k_4 h\right) \\ k_6 &= f\left(x_i + h, y_i - \frac{3}{7}k_1 h + \frac{2}{7}k_2 h + \frac{12}{7}k_3 h - \frac{12}{7}k_4 h + \frac{8}{7}k_5 h\right) \\ y_{i+1} &= y_i + \frac{1}{90}h(7k_1 + 32k_3 + 12k_4 + 32k_5 + 7k_6) \end{aligned} \tag{3.60}$$

3.6.8. Método de Runge-Kutta Extendido

Se hace una extensión del método RK para ecuaciones diferenciales ordinarias de segundo orden de la forma:

$$y'' = \frac{d^2y}{dx^2} = f(x, y, y') \tag{3.61}$$

En la cual utilizando un adecuado cambio de variable podemos establecer:

$$z = y' \ldots\ldots\ldots\ldots z' = y'' \tag{3.62}$$

La cual, evidentemente, ha sido transformada, y representa una EDO de primer orden. De esta manera el sistema de ecuaciones queda reducido a:

$$y' = f(x, y, z) \tag{3.63}$$
$$z' = g(x, y, z) \tag{3.64}$$

Para dichas ecuaciones se requiere determinar los siguientes coeficientes:

$$
\begin{aligned}
k_1 &= hf(x_o, y_o, z_o) \\
L_1 &= hg(x_o, y_o, z_o) \\
k_2 &= hf\left(x_o + \frac{h}{2}, y_o + \frac{k_1}{2}, z_o + \frac{L_1}{2}\right) \\
L_2 &= hg\left(x_o + \frac{h}{2}, y_o + \frac{k_1}{2}, z_o + \frac{L_1}{2}\right) \\
k_3 &= hf\left(x_o + \frac{h}{2}, y_o + \frac{k_2}{2}, z_o + \frac{L_2}{2}\right) \\
L_3 &= hg\left(x_o + \frac{h}{2}, y_o + \frac{k_2}{2}, z_o + \frac{L_2}{2}\right) \\
k_4 &= hf(x_o + h, y_o + k_3, z_o + L_3) \\
L_4 &= hg(x_o + h, y_o + k_3, z_o + L_3)
\end{aligned}
\tag{3.65}
$$

$$
y_1 = y_o + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{3.66}
$$

$$
z_1 = z_o + \frac{1}{6}(L_1 + 2L_2 + 2L_3 + L_4) \tag{3.67}
$$

3.6.9. Método del Punto Medio

Método en el que se establece el siguiente sistema:

$$
\begin{aligned}
k_1 &= f(x_i, y_i) \\
k_2 &= f\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}hk_1\right) \\
y_{i+1} &= y_i + k_2 h
\end{aligned}
\tag{3.68}
$$

Se observa que k_1 es la pendiente en el inicio del intervalo y k_2 la pendiente en el punto medio del mismo. Entonces, con estos valores de constantes se reproduce el método del punto medio.

3.6.10. Método de Ralston

Método en el que se establece el siguiente sistema:

$$
\begin{aligned}
k_1 &= f(x_i, y_i) \\
k_2 &= f\left(x_i + \frac{3}{4}h, y_i + \frac{3}{4}hk_1\right) \\
y_{i+1} &= y_i + h\left(\frac{1}{3}k_1 + \frac{2}{3}k_2\right)
\end{aligned}
\tag{3.69}
$$

Se observa que k_1 es la pendiente en el inicio del intervalo y k_2 la pendiente en el punto ubicado a 3/4 del mismo. Este es el denominado método de Ralston.

3.6.11. Métodos Runge-Kutta-Fehlberg de 2.° Orden de Paso Variable

Los métodos del tipo Runge-Kutta no tienen forma de evaluar el error cometido en cada iteración; sin embargo, es posible combinar dos métodos de orden diferente, o utilizar un método con dos pasos de integración diferentes para estimar el error y, en base a este, decidir si es necesario o no cambiar el tamaño del paso de integración (reducirlo o incrementarlo), teniéndose, entonces, un método de paso variable; por ejemplo, los métodos Runge-Kutta-Fehlberg son métodos del tipo Runge-Kutta en los cuales se combinan dos métodos de diferente orden para poder estimar el error cometido en cada iteración de la solución.

Este método está dado por las siguientes ecuaciones:

$$k_1 = hf(y_n, t_n) \tag{3.70}$$
$$k_2 = hf\left(y_n + \frac{1}{2}k_1, t_n + \frac{1}{2}h\right)$$
$$k_3 = hf\left(y_n + \frac{1}{256}k_1 + \frac{255}{256}k_2, t_n + h\right)$$
$$y_{n+1} = y_n + \frac{1}{512}k_1 + \frac{255}{256}k_2 + \frac{1}{512}k_3$$

3.6.12. Métodos Multipaso

Se pueden diseñar procedimientos más eficientes si se utilizan valores previos para el cálculo de los posteriores; en este principio se basan los métodos multipaso. A continuación, exponemos los métodos multipaso.

3.6.12.1. Método de Adams-Bashforth de 5.º Orden. Considerando una EDO con valor inicial conocido, este método establece:

$$y_{n+1} = y_n + \frac{h}{720}[1901f(x_n, y_n) - 2774f(x_{n-1}, y_{n-1}) \tag{3.71}$$
$$+ 2616f(x_{n-2}, y_{n-2}) - 1274f(x_{n-3}, y_{n-3}) + 251f(x_{n-4}, y_{n-4})]$$

Es evidente la necesidad de contar con los valores de $f(x_0,y_0)$, los cuales se obtienen con las condiciones iniciales $f(x_1,y_1)$, $f(x_2,y_2)$, $f(x_3,y_3)$, $f(x_4,y_4)$. Estos términos se obtienen a partir de un método de solo un paso, como, por ejemplo, algún método Runge-Kutta.

3.6.12.2. Método de Adams-Bashforth de 4.º Orden. Considerando una EDO con valor inicial conocido, y el procedimiento similar al anterior:

$$y_{n+1} = y_n + \frac{h}{24}[55f(x_n, y_n) - 59f(x_{n-1}, y_{n-1}) \tag{3.72}$$
$$+ 37f(x_{n-2}, y_{n-2}) - 9f(x_{n-3}, y_{n-3})]$$

3.6.12.3. Método de Adams-Moulton de 5.º Orden. En la práctica, las fórmulas de Adams-Bashforth no se utilizan de manera aislada, se usan junto a otras fórmulas para aumentar la precisión. Una de ellas es la de Adams-Moulton de 5.º orden.

$$y_{n+1} = y_n + \frac{h}{720}[251f(x_{n+1}, y_{n+1}) + 646f(x_n, y_n) - 19f(x_{n-3}, y_{n-3})] \quad - 246f(x_{n-1}, y_{n-1}) + 106f(x_{n-2}, y_{n-2}) \tag{3.73}$$

Es evidente que esta fórmula no puede emplearse como método directo para el cálculo de la solución, debido a que a la derecha de la expresión aparece el término $f(x_{n+1}, y_{n+1})$. Un interesante algoritmo es el llamado predictor-corrector, este emplea la fórmula de Adams-Bashforth para predecir un valor tentativo de y_{n+1}, al cual podemos llamar y^*_{n+1}, y luego se recurre a la fórmula de Adams-Moulton para calcular el valor corregido de y_{n+1}. Así en la fórmula de Adams-Moulton el valor de $f(x_{n+1}, y_{n+1})$ se calcula como $f(x_{n+1}, y^*_{n+1})$.

Finalmente, el predictor es:

$$y^*_{n+1} = y_n + \frac{h}{720}[1901f(x_n, y_n) - 2774f(x_{n-1}, y_{n-1}) + 251f(x_{n-4}, y_{n-4})] + 2616f(x_{n-2}, y_{n-2}) - 1274f(x_{n-3}, y_{n-3}) \tag{3.74}$$

Y el corrector es:

$$y_{n+1} = y_n + \frac{h}{720}[251f(x_{n+1}, y^*_{n+1}) + 646f(x_n, y_n) - 19f(x_{n-3}, y_{n-3})] - 246f(x_{n-1}, y_{n-1}) + 106f(x_{n-2}, y_{n-2}) \tag{3.75}$$

3.6.12.4. Método de Adams-Moulton de 4.º Orden. Siguiendo el mismo procedimiento, las fórmulas recursivas son:

El predictor es:

$$y^*_{n+1} = y_n + \frac{h}{24}[55f(x_n, y_n) - 59f(x_{n-1}, y_{n-1}) + 37f(x_{n-2}, y_{n-2}) - 9f(x_{n-3}, y_{n-3})] \tag{3.76}$$

Y el corrector es:

$$y_{n+1} = y_n + \frac{h}{24}[9f(x_{n+1}, y^*_{n+1}) + 19f(x_n, y_n) - 5f(x_{n-1}, y_{n-1}) + f(x_{n-2}, y_{n-2})] \tag{3.77}$$

3.6.12.5. Método de Milne. Es otro método multipaso, pero que posee problemas de estabilidad.

El predictor es:

$$y_{n+1}^{*} = y_{n-3} + \frac{4h}{3}[2f(x_n, y_n) - f(x_{n-1}, y_{n-1}) + 2f(x_{n-2}, y_{n-2})] \quad (3.78)$$

Y el corrector es:

$$y_{n+1} = y_{n-1} + \frac{h}{3}[f(x_{n+1}, y_{n+1}^{*}) + 4f(x_n, y_n) + f(x_{n-1}, y_{n-1})] \quad (3.79)$$

De igual manera que en el método de Adams-Moulton se debe usar un método de un paso para calcular los primeros valores, preferentemente un método RK.

3.6.13. ODE en MATLAB

MATLAB dispone de varias funciones para resolver, mediante procedimientos numéricos, ecuaciones diferenciales, siendo una de las más difundidas la *ode45*.

ode45, ode23, ode113, ode15s, ode23s, ode23t, ode23tb

La función *ode45*, tiene la siguiente sintaxis:

[t,x]=*ode45*(odefun,tspan,x0, options, params)

- "x" es una matriz donde cada columna corresponde a las variables dependientes y t es el vector tiempo.
- "odefun" es el nombre de la función.
- "tspan" especifica el intervalo de tiempo, un vector de dos números tspan=[ti,tf], tiempo inicial y final. Para obtener valores de las variables dependientes en instantes concretos t_0, t_1, t_2... t_n. Se escribe tspan= [t_0, t_1...t_n].
- "x_0" es un vector que contiene los valores iniciales.
- "options" es una estructura que se crea con la función *odeset*.
- "params" son parámetros que queremos pasar a la función *odefun*.

En la mayor parte de los ejemplos podemos utilizar los tres primeros parámetros: llamaremos a la función *ode45* y le pasaremos la función *odefunc*, los instantes inicial y final en el vector *tspan* y las condiciones iniciales en el vector x_0.

3.7. Estudio de Casos

3.7.1. Tiempo de Vaciado de un Recipiente Cilíndrico Vertical

Formulación del caso:

Un tanque cilíndrico de fondo plano con un diámetro de D=1.5 m contiene un líquido de densidad 1.0 Kg/L, a una altura a de 3 m. Se desea saber la altura del líquido dentro del tanque 3 minutos después de que se abre completamente la válvula de salida, la cual da un gasto de Q_s=$0.6\,A(2gh)^{1/2}$, donde A es el área seccional del tubo de salida con diámetro d, y es de 78.5 10^{-4} m^2 y g=9.81 m/s^2.

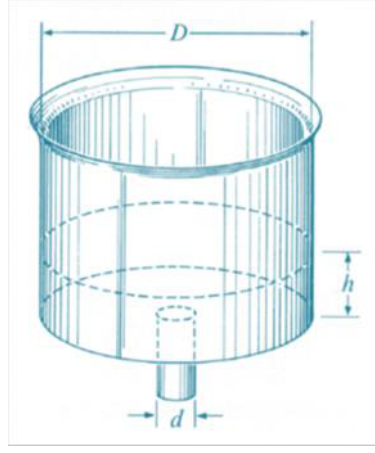


Figura 3.3.
Recipiente cilíndrico vertical.

Solución:

Para el caso de llenado o vaciado de un tanque cilíndrico se modela haciendo un balance de materia con la siguiente expresión universal:

$$Acumulacion = Entradas - Salidas \tag{3.80}$$

$$\frac{d(V\rho)}{dt} = 0 - 0{,}6A\sqrt{2gh} \tag{3.81}$$

$$V = \frac{\Pi}{4}\phi^2 h \tag{3.82}$$

$$\frac{\Pi}{4}\phi^2\frac{dh}{dt} = -0{,}6A\sqrt{2gh} \tag{3.83}$$

$$\frac{dh}{dt} = -0{,}0026653\sqrt{2gh} \tag{3.84}$$

El valor cero en la ecuación diferencial, como primer valor del lado derecho de la ecuación, representa que en el sistema no hay ingreso de fluidos. Al considerar como tiempo cero al abrir la válvula y, además, la altura buscada a un tiempo de 180 s, se tiene el siguiente sistema a resolver:

$$\begin{cases} \dfrac{dh}{dt} = -0{,}0026653\sqrt{2gh} \\ h(0) = 3 \\ h(180) = ? \end{cases} \tag{3.85}$$

Establecida la ecuación diferencial ordinaria EDO, el valor inicial y el requerimiento de cálculo, podemos utilizar varios métodos de solución; los mismos que proponemos a continuación:

Solución Analítica:

La ecuación diferencial anterior es de fácil integración analítica, por lo tanto, tenemos la siguiente solución:

$$\int dt = \frac{1}{-0{,}0026653\sqrt{2g}} \int h^{-0{,}5}\, dh \quad -84{,}7041 \int h^{-0{,}5}\, dh \tag{3.86}$$

La integral de la ecuación resulta:

$$t = -169{,}4082\, h^{0{,}5} + C \tag{3.87}$$

La constante de integración la calculamos usando: t=0, h=3 m.
Por tanto: 0=(- 169.4082) x $(3)^{0{,}5} + C$ …….. C=293.4236

Finalmente tenemos:

$$t = -169{,}4082\, h^{0{,}5} + 293{,}4236 \tag{3.88}$$

$$h = \left[\frac{(293{,}4236 - t)}{169{,}4028}\right]^2 \tag{3.89}$$

Esta ecuación resultante la tabulamos obteniéndose:

Solución Analítica			
t(s)	h(m)	t(s)	h(m)
0	2.9999	150	0.7167
10	2.7989	160	0.6203
20	2.6049	170	0.5308
30	2.4179	180	0.4482
40	2.2378	190	0.3727
50	2.0646	200	0.3041
60	1.8985	210	0.2425
70	1.7393	220	0.1878
80	1.5871	230	0.1401
90	1.4418	240	0.0994
100	1.3036	250	0.0657
110	1.1723	260	0.0389
120	1.0479	270	0.0191
130	0.9306	280	0.0063
140	0.8202	290	0.0004

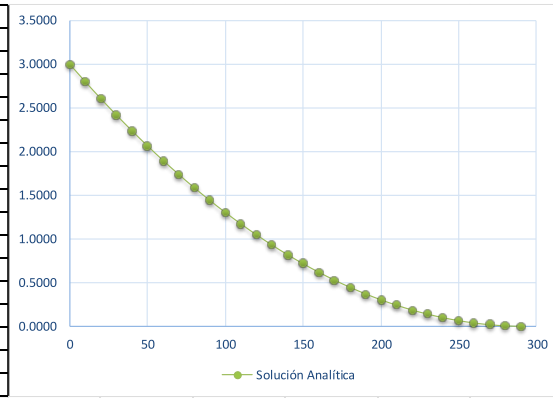


Figura 3.4.
Vaciado del cilindro vertical para la solución analítica.

Respuesta: 0.4482 m a los 3 min o 180 s.

Método de Euler:

Usando las ecuaciones del método de Euler para este problema de valor inicial y haciendo la formulación en hoja de cálculo, la solución es la siguiente:

Método de Euler				
delta t=	10		delta t=	20
	Dh=10 s			Dh=20 s
t(s)	h(m)		t(s)	h(m)
0	3		0	3
10	2.7955		20	2.5910
20	2.5981		40	2.2110
30	2.4078		60	1.8599
40	2.2246		80	1.5379
50	2.0486		100	1.2451
60	1.8796		120	0.9816
70	1.7177		140	0.7477
80	1.5630		160	0.5435
90	1.4154		180	0.3694
100	1.2749			
110	1.1416			
120	1.0155			
130	0.8965			
140	0.7847			
150	0.6802			
160	0.5828			
170	0.4927			
180	0.4098			

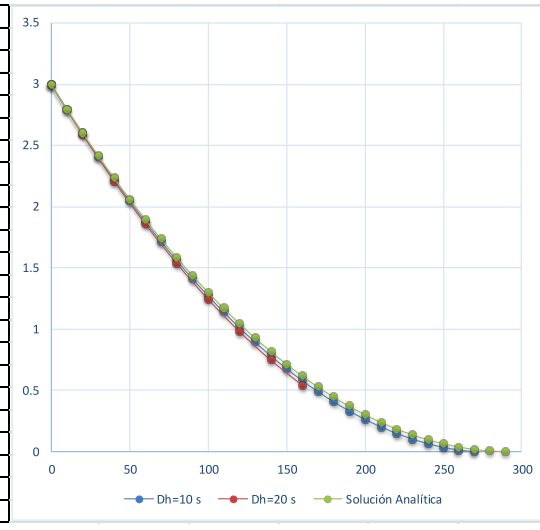


Figura 3.5.
Vaciado del cilindro vertical para la solución de Euler.

Respuesta: 0.4098 m (dt=10 s) y 0.3694 m (dt=20 s) a los 3 min o 180 s.

La gráfica muestra una diferencia poco significativa entre el método analítico y la aplicación del método de Euler para pasos de 10 y 20 s.

A continuación, se presenta un código computacional en MATLAB para la solución del caso usando el método de Euler.

```
%METODO DE EULER
clc, clear;
h=10;
tf=180; %tiempo final de calculo
b=tf/h; %numero de pasos
y(1)=3;
for  j=1:b
     y(j+1)=y(j)-h*(0.0026653*(2*9.81*y(j))^0.5)
     [y,j]
end
r=0:10:180
plot(r,y,'r-')%,t,y,'-bs')
```

```
xlabel('TIEMPO DE VACIADO(SEG)')
ylabel('ALTURAS EN EL TANQUE(m)')
title('CURVA DE TIEMPO DE VACIADO METODO EULER')
h1 = legend('Alturas',1);
```

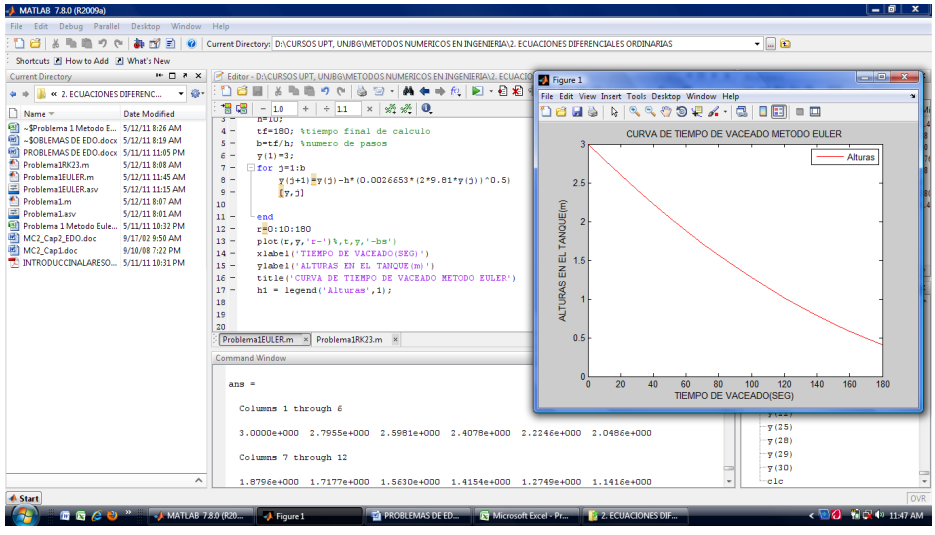


Figura 3.6.
Salida del programa para el tiempo de vaciado del método Euler

Respuesta: 0.4098 m a los 3 min.

Euler Mejorado:

Establecida la ecuación diferencial ordinaria EDO, el valor inicial y el requerimiento de cálculo, también podemos utilizar el método de Euler mejorado.

$$y_{n+1} = y_n + \frac{h}{2}\left[f(x_n, y_n) + f(x_n + h, y_n + h \cdot f(x_n, y_n))\right] \qquad (3.90)$$

$$y_{n+1} = y_n + \frac{h}{2}\left[f(y_n) + f(y_n + h \cdot f(y_n))\right]$$

Como la función solo tiene variabilidad en "y", como altura en el tanque de agua, siendo "h" el paso de tiempo para el cálculo. La solución de la ecuación anterior se puede programar en MATLAB de la siguiente manera:

```
%METODO DE EULER MEJORADO
clc, clear;
h=10;
tf=180; %tiempo final de calculo
b=tf/h; %numero de pasos
y(1)=3;
for  j=1:b
```

```
        yp(j)=-0.0026653*(2*9.81*(y(j)+h*(-0.0026653*(2*9.81*(y(j))^0.5))));
        y(j+1)=y(j)+(h/2)*(((-0.0026653*(2*9.81*y(j))^0.5)+yp(j)));

end
[y,j]
r=0:10:180;
plot(r,y,'r-')%,t,y,'-bs')
xlabel('TIEMPO DE VACIADO(SEG)')
ylabel('ALTURAS EN EL TANQUE(m)')
title('CURVA DE TIEMPO DE VACEADO METODO EULER MEJORADO')
h1 = legend('Alturas',1);
```

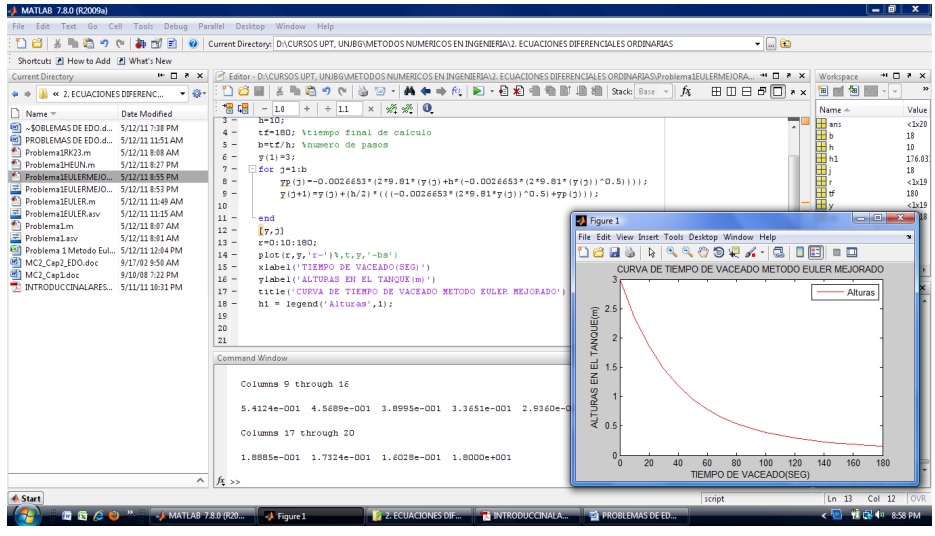


Figura 3.7.
Salida del programa para el tiempo de vaciado del método Euler mejorado

Respuesta: 0.1602 m a los 3 min.

Método de Heun o Predictor-Corrector:

También se presenta un código computacional en MATLAB para la solución del caso usando el método de Heun o Predictor-Corrector.

```
%METODO DE HEUN O PREDICTOR-CORRECTOR
clc, clear;
h=10;
tf=180; %tiempo final de calculo
b=tf/h; %numero de pasos
y(1)=3;
for  j=1:b
     yp(j+1)=y(j)-h*((0.0026653*(2*9.81*y(j))^0.5));
     y(j+1)=y(j)-
(h/2)*((0.0026653*(2*9.81*y(j))^0.5)+0.0026653*(2*9.81*(yp(j+1)))^0.5);
```

```
end
[y,j]
r=0:10:180;
plot(r,y,'r-')%,t,y,'-bs')
xlabel('TIEMPO DE VACIADO(SEG)')
ylabel('ALTURAS EN EL TANQUE(m)')
title('CURVA DE TIEMPO DE VACIADO METODO HEUN O PREDICTOR-CORRECTOR')
h1 = legend('Alturas',1);
```

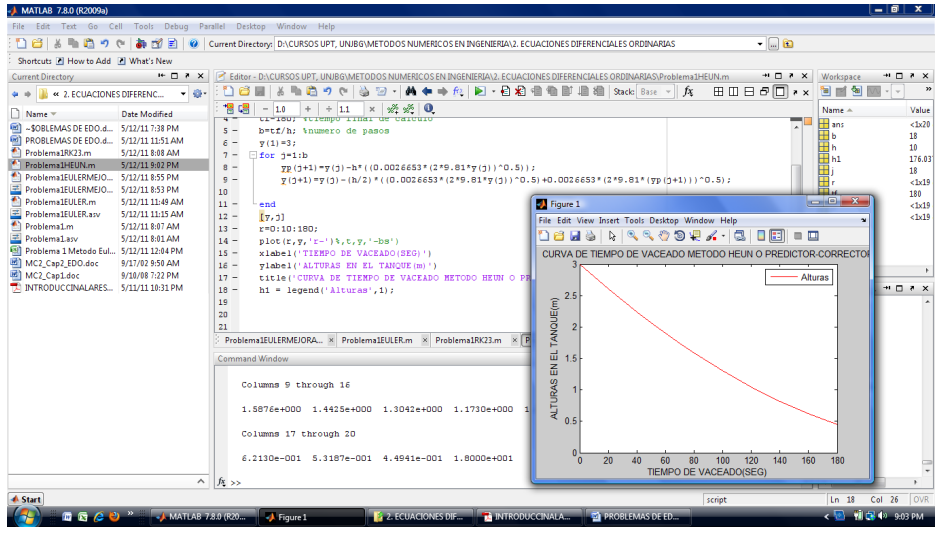


Figura 3.8.
Salida del programa para el tiempo de vaciado del método Heun o Predictor-Corrector

Respuesta: 0.4941 m a los 3 min.

Runge-Kutta 23.

Adicionalmente, se presenta un código computacional en MATLAB para la solución del caso usando el método de Runge-Kutta 23.

```
%METODO DE RUNGE KUTTA 23
clc, clear;
f=inline('-0.0026653*(2*9.81*y)^0.5','t','y');
[t,y]=ode23(f,[0,10,180],3);
[t,y]
plot(t,y,'r-')%,t,y,'-bs')
xlabel('TIEMPO DE VACIADO(SEG)')
ylabel('ALTURAS EN EL TANQUE(m)')
title('CURVA DE TIEMPO DE VACIADO METODO RK-23')
h1 = legend('Alturas',1);
```

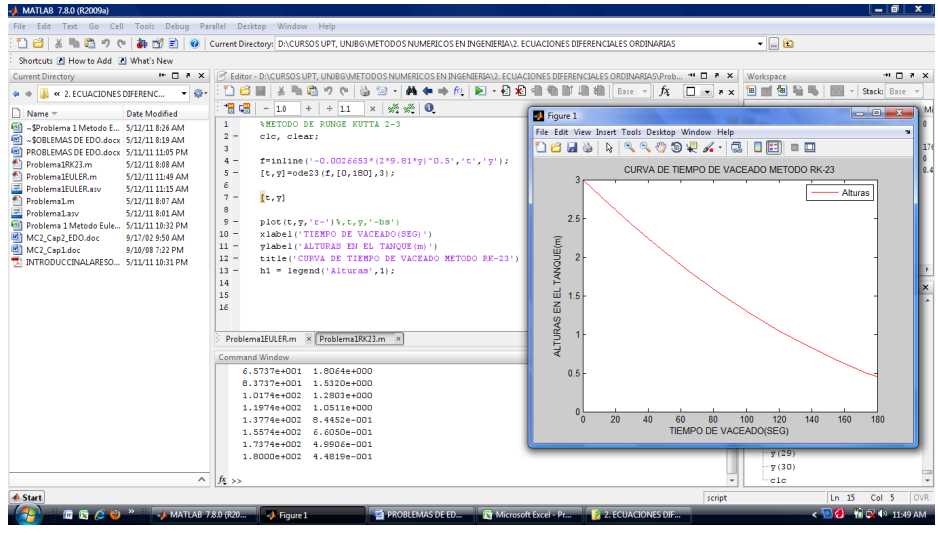


Figura 3.9.
Salida del programa para el tiempo de vaciado del método RK 23

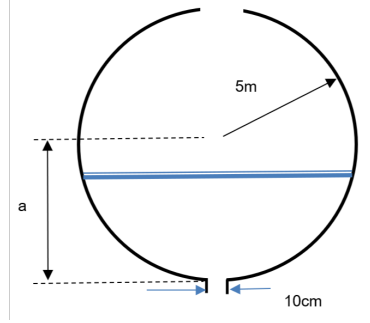
Respuesta: 0.4482 m a los 3 min.
Resumen de resultados para el vaciado de un tanque:

Método utilizado	**Delta de "t" (s)**	**Tiempo final (s)**	**Valor "h" final (m)**
Solución analítica	10	180	0.4482
Euler	10	180	0.4098
Euler mejorado	10	180	0.1602
Heun	10	180	0.4941
Runge-Kutta 23	10	180	0.4482

3.7.2. Tiempo de Vaciado de un Recipiente Esférico

Formulación del caso:

Calcule el tiempo necesario para que el nivel del líquido dentro del tanque esférico (Chapra, 2017), con radio r=5 m mostrado en la figura, descienda de 4 a 3 m. La velocidad de salida por el orificio del fondo es v=4.895$(a)^{1/2}$, el diámetro de dicho orificio es de 10 cm.



Solución:

Para el caso de llenado o vaciado de un tanque esférico se modela haciendo un balance de materia con la siguiente expresión universal:

$$Acumulacion = Entradas - Salidas \tag{3.91}$$

$$\frac{d(V\rho)}{dt} = 0 - Av\rho$$

$$V = \pi(5a^2 - \frac{a^2}{3}) \tag{3.92}$$

$$A = \frac{\pi}{4}\phi^2 = \frac{\pi}{4}(0{,}10)^2 \tag{3.93}$$

$$v = 4{,}895(a)^{1/2} \tag{3.94}$$

$$\pi\frac{d(5a^2 - \frac{a^2}{3})}{dt} = -\frac{\pi}{4}(0{,}10)^2 . 4{,}895(a)^{1/2} \tag{3.95}$$

$$\frac{da}{dt} = -\frac{4{,}895(0{,}10)^2(a)^{1/2}}{4(10a - a^2)} \tag{3.96}$$

Al considerar como tiempo cero al abrir la válvula, y además la altura buscada a un tiempo en el cual el nivel en el depósito desciende de 4 a 3 m, el sistema a resolver es:

$$\begin{cases} \frac{da}{dt} = -\frac{0{,}122375\,(a)^{1/2}}{(10a - a^2)} \\ a(0) = 4 \\ a(?) = 3 \end{cases} \tag{3.97}$$

Establecida la EDO, el valor inicial y el requerimiento de cálculo, podemos utilizar varios métodos de solución, los mismos que proponemos a continuación:

Solución Analítica:

La ecuación diferencial propuesta es factible de integración analítica, por lo tanto, tenemos la siguiente solución utilizando Wolfram Mathematica (*on line*):

$$\int dt = \int \frac{(a^2 - 10a)}{0{,}122375(a)^{1/2}}\, da \tag{3.98}$$

$$t = 3{,}26864\, a^{2{,}5} - 54{,}4774\, a^{1{,}5} + C$$

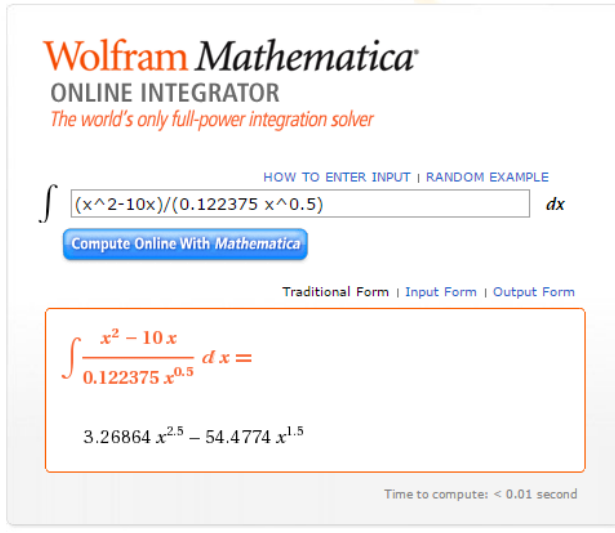


Figura 3.10.
Integración con Wolfram Mathematica, online integrator

La constante de integración la calculamos usando: a=4, t=0.
Por tanto: 0= $3{,}26864\ (4)^{2{,}5} - 54{,}4774\ (4)^{1{,}5} + C$ ..... C=+331.2227

Finalmente, tenemos:

$$t = 3{,}26864\ a^{2{,}5} - 54{,}4774\ a^{1{,}5} + 331{,}2227 \qquad (3.99)$$

Esta ecuación resultante la tabulamos, obteniéndose:

Solución Analítica			
t(s)	a(m)	t(s)	a(m)
0.000	4.000	74.279	3.250
4.908	3.950	79.250	3.200
9.825	3.900	84.219	3.150
14.752	3.850	89.185	3.100
19.686	3.800	94.146	3.050
24.628	3.750	99.103	3.000
29.576	3.700	104.054	2.950
34.530	3.650	108.997	2.900
39.489	3.600	113.933	2.850
44.453	3.550	118.861	2.800
49.420	3.500	123.778	2.750
54.389	3.450	128.685	2.700
59.361	3.400	133.580	2.650
64.334	3.350	138.462	2.600
69.307	3.300	143.330	2.550

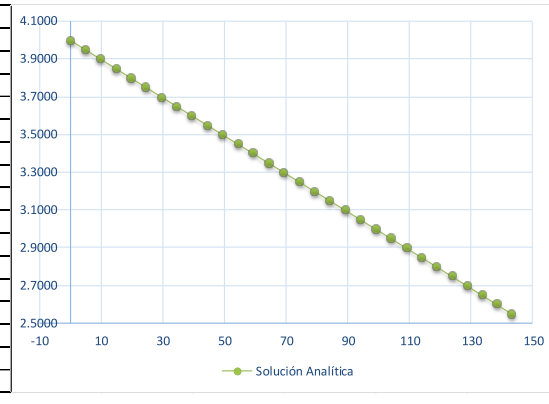


Figura 3.11.
Solución analítica para el tiempo de descarga de la parte inferior del recipiente esférico

Respuesta: 99.103 s para bajar de 4 a 3 m.

Método de Euler:

Según las ecuaciones planteadas para el método de Euler en este problema de valor inicial, se formuló una hoja de cálculo obteniéndose la siguiente solución:

Método de Euler				
delta t=	10		delta t=	20
	Dh=10 s			Dh=20 s
t(s)	a(m)		t(s)	a(m)
0	4.0000		0	4.0000
10	3.8980		20	3.7960
20	3.7964		40	3.5936
30	3.6952		60	3.3920
40	3.5942		80	3.1909
50	3.4935		100	2.9897
60	3.3928		120	2.7878
70	3.2923		140	2.5845
80	3.1917		160	2.3792
90	3.0911		180	2.1710
100	2.9904			
110	2.8894			
120	2.7882			
130	2.6866			
140	2.5845			
150	2.4818			
160	2.3785			
170	2.2744			
180	2.1693			

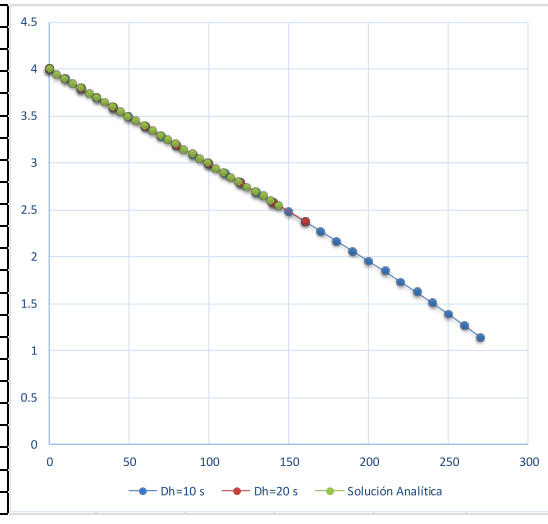


Figura 3.12.
Solución analítica para el tiempo de descarga de la parte inferior del recipiente esférico

Respuesta: 100 s para un descenso a 2.9904 m con dt=10 s y 100 s con dt=20 s para descender a 2.9897 m el nivel en el depósito. La gráfica muestra una diferencia poco significativa entre el método analítico y la aplicación del método de Euler para pasos de 10 y 20 s. Es notorio que el cambio de tamaño de paso varía la aproximación en los resultados obtenidos.

A continuación, se presenta un código computacional en MATLAB para la solución del caso usando el método de Euler.

```
%METODO DE EULER
clc, clear;
h=10;
tf=120; %tiempo final de calculo
b=tf/h; %numero de pasos
y(1)=4;
for   j=1:b
     y(j+1)=y(j)-h*((0.122375*(y(j))^0.5)/(10*y(j)-
(y(j))^2))
end
[y,j]
r=0:10:120;
plot(r,y,'r-')%,t,y,'-bs')
```

```
grid
xlabel('TIEMPO DE VACEADO(SEG)')
ylabel('ALTURAS EN EL TANQUE(m)')
title('CURVA DE TIEMPO DE VACEADO METODO EULER')
h1 = legend('Alturas',1);
```

La salida del programa es la siguiente:

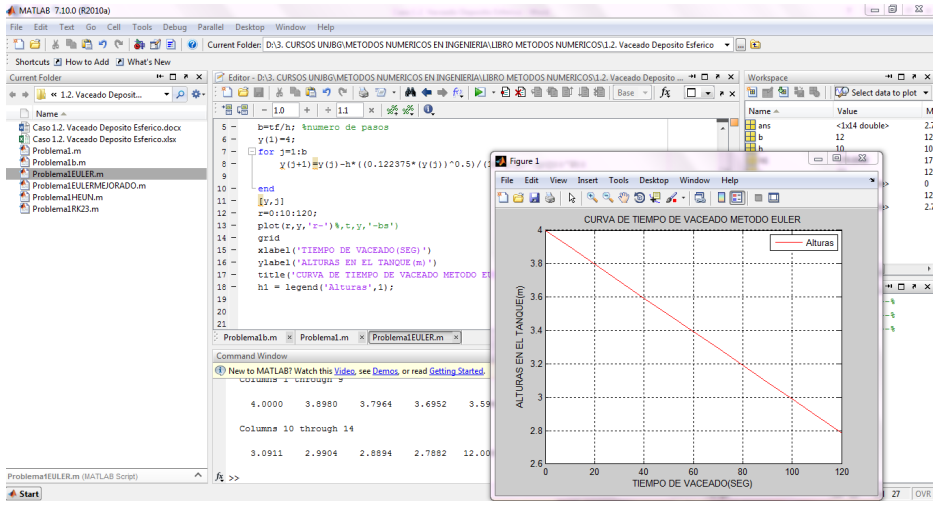


Figura 3.13.
Solución de Euler en MATLAB para el tiempo de descarga de la parte inferior del recipiente esférico

Respuesta: 2.9904 m a los 100 s.

Euler mejorado:

Establecida la EDO, el valor inicial y el requerimiento de cálculo, también podemos utilizar el método de Euler mejorado según el siguiente programa MATLAB.

```
%METODO DE EULER MEJORADO
clc, clear;
h=10;
tf=100; %tiempo final de calculo
b=tf/h; %numero de pasos
y(1)=4;
for  j=1:b
     yp(j)=((-0.122375*(y(j))^0.5)/(10*y(j)-(y(j))^2));
     y(j+1)=y(j)+(h/2)*((-(0.122375*(y(j))^0.5)/(10*y(j)-(y(j))^2)+yp(j)));
end
y,j]
r=0:10:tf;
plot(r,y,'r-')%,t,y,'-bs')
```

```
grid
xlabel('TIEMPO DE VACEADO(SEG)')
ylabel('ALTURAS EN EL TANQUE(m)')
title('CURVA DE TIEMPO DE VACIADO METODO EULER MEJORADO')
h1 = legend('Alturas',1);
```

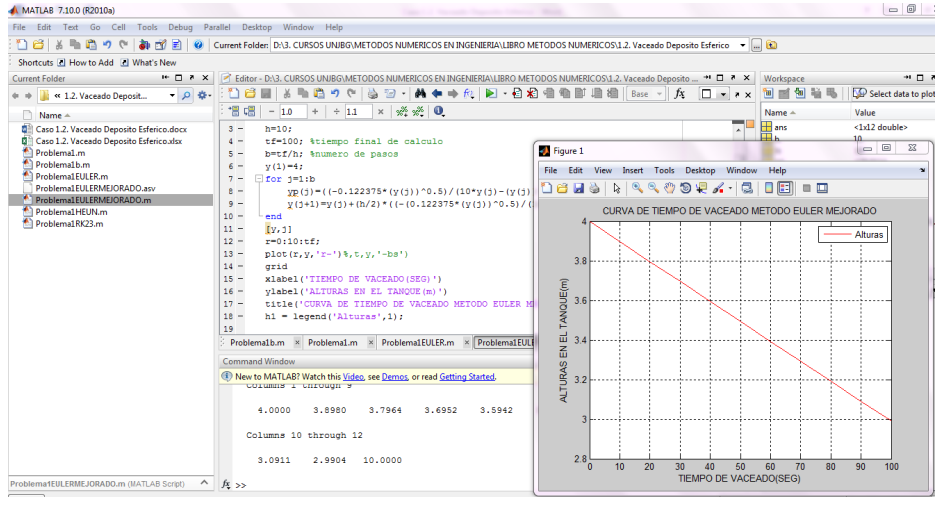


Figura 3.14.
Solución de Euler en MATLAB para el tiempo de descarga de la parte inferior del recipiente esférico

Respuesta: 2.9904 m a los 100 s.

Método de Heun o Predictor-Corrector:

También se presenta un código computacional en MATLAB para la solución del caso usando el método de Heun o Predictor-Corrector. Se obtiene como respuesta: 2.9909 m de descenso a los 100 s.

```
%METODO DE HEUN O PREDICTOR-CORRECTOR
clc, clear;
h=10;
tf=100; %tiempo final de calculo
b=tf/h; %numero de pasos
y(1)=4;
for  j=1:b
     yp(j+1)=y(j)+h*((-0.122375*(y(j))^0.5)/(10*y(j)-(y(j))^2));
     y(j+1)=y(j)+(h/2)*((-0.122375*(y(j))^0.5)/(10*y(j)-(y(j))^2))+...
     (-0.122375*(yp(j+1))^0.5)/(10*yp(j+1)-(yp(j+1))^2));
end
[y,j]
r=0:10:tf;
plot(r,y,'r-')%,t,y,'-bs')
grid
```

```
xlabel('TIEMPO DE VACIADO(SEG)')
ylabel('ALTURAS EN EL TANQUE(m)')
title('CURVA DE TIEMPO DE VACIADO METODO HEUN O PREDICTOR-CORRECTOR')
h1 = legend('Alturas',1);
```

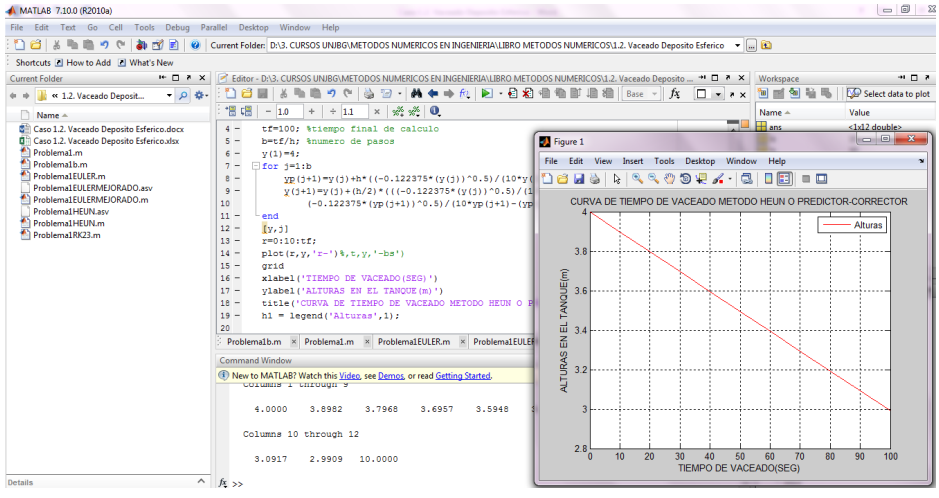


Figura 3.15.
Solución de Heun para el tiempo de descarga de la parte inferior del recipiente esférico

Runge-Kutta 23:

Adicionalmente, se presenta un código computacional en MATLAB para la solución del caso, usando el método de Runge-Kutta 23. Se obtiene como respuesta: 3 m a los 99.1 s.

```
%METODO DE RUNGE KUTTA 23
clc, clear;
f=inline('(-0.122375*y^0.5)/(10*y-y^2)','t','y');
[t,y]=ode23(f,[0,99.1],4);
[t,y]
plot(t,y,'r-')%,t,y,'-bs')
xlabel('TIEMPO DE VACEADO(SEG)')
ylabel('ALTURAS EN EL TANQUE(m)')
title('CURVA DE TIEMPO DE VACEADO METODO RK-23')
h1 = legend('Alturas',1);
```

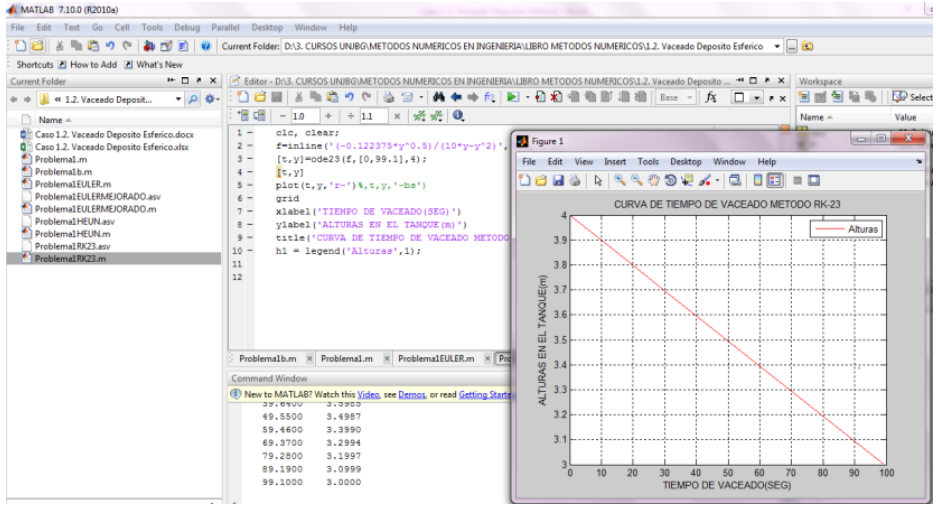


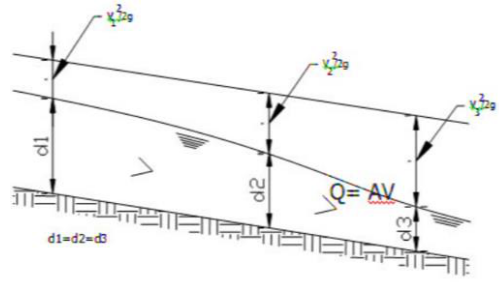
Figura 3.16.
Solución de Runge-Kutta 23 para el tiempo de descarga de la parte inferior del recipiente esférico

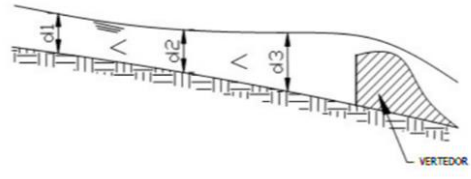
3.7.3. Cálculo de un Eje Hidráulico o Curva de Remanso

Formulación del caso:

Calcular el eje hidráulico o curva de remanso que se genera a partir de un barraje aguas arriba si se tiene los siguientes datos:

Caudal	Q	=	5 m³/s
Rugosidad	n	=	0.025
Talud	Z	=	1.5
Ancho Base	b	=	2.5 m
Pendiente	So	=	0.0005
Y inicial	Yi	=	2.3 m
Y final	Yf	=	1.4 m





Se debe utilizar la ecuación dinámica del flujo gradualmente variado para calcular los tirantes de agua en función del espacio, combinada con la ecuación para flujo uniforme conocida como ecuación de Manning.

$$\frac{dy}{dx} = So\left[\frac{1-\frac{Se}{So}}{1-\frac{Q^2T}{gA^3}}\right] \qquad (3.100)$$

$$Q = \frac{1}{n} A R^{2/3} S^{1/2} \tag{3.101}$$

Solución:

La ecuación planteada debe ser reestructurada de tal forma que se despeje "dy" como variable a ser calculada en función de "dx" de la siguiente manera:

$$dy = \left[\frac{So - Se}{1 - \frac{Q^2 T}{g A^3}}\right] dx \tag{3.102}$$

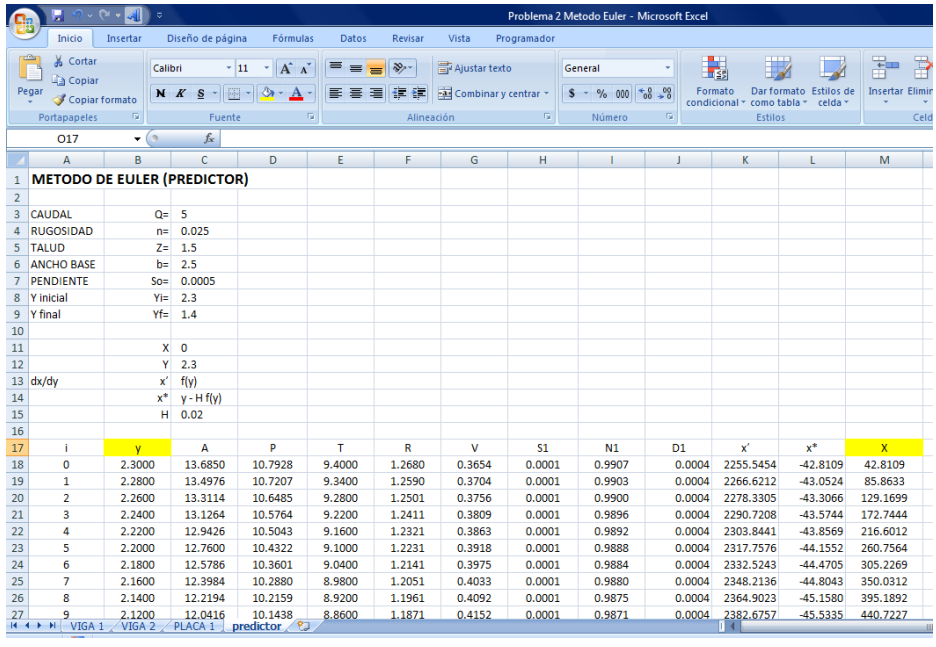
$$Se = \left(\frac{Qn}{AR^{2/3}}\right)^2 \tag{3.103}$$

So = Es la pendiente de fondo del canal
n = Coeficiente de rugosidad
A = Es el área hidráulica = $by+zy^2$
P = $b+2y(1+z^2)^{1/2}$
$R=A/P$ = Radio hidráulico
T = Espejo de agua = $b+2zy$

Establecida la EDO, el valor inicial y el requerimiento de cálculo, podemos utilizar el método de Euler, en virtud a que la exactitud de los resultados esperados o requeridos no es grande. Por lo tanto, el problema queda estructurado matemáticamente como:

$$\begin{cases} \frac{dy}{dx} = \left[\frac{So - Se}{1 - \frac{Q^2 T}{g A^3}}\right] \\ y(0) = 2.3 \\ y(x = ?) = 1.4 \end{cases} \tag{3.104}$$

La formulación en hoja de cálculo para la solución es la siguiente:



Problema 2 Metodo Euler - Microsoft Excel

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	**METODO DE EULER (PREDICTOR)**												
2													
3	CAUDAL	Q=	5										
4	RUGOSIDAD	n=	0.025										
5	TALUD	Z=	1.5										
6	ANCHO BASE	b=	2.5										
7	PENDIENTE	So=	0.0005										
8	Y inicial	Yi=	2.3										
9	Y final	Yf=	1.4										
10													
11		X	0										
12		Y	2.3										
13	dx/dy	x´	f(y)										
14		x*	y - H f(y)										
15		H	0.02										
16													
17	i	y	A	P	T	R	V	S1	N1	D1	x´	x*	X
18	0	2.3000	13.6850	10.7928	9.4000	1.2680	0.3654	0.0001	0.9907	0.0004	2255.5454	-42.8109	42.8109
19	1	2.2800	13.4976	10.7207	9.3400	1.2590	0.3704	0.0001	0.9903	0.0004	2266.6212	-43.0524	85.8633
20	2	2.2600	13.3114	10.6485	9.2800	1.2501	0.3756	0.0001	0.9900	0.0004	2278.3305	-43.3066	129.1699
21	3	2.2400	13.1264	10.5764	9.2200	1.2411	0.3809	0.0001	0.9896	0.0004	2290.7208	-43.5744	172.7444
22	4	2.2200	12.9426	10.5043	9.1600	1.2321	0.3863	0.0001	0.9892	0.0004	2303.8441	-43.8569	216.6012
23	5	2.2000	12.7600	10.4322	9.1000	1.2231	0.3918	0.0001	0.9888	0.0004	2317.7576	-44.1552	260.7564
24	6	2.1800	12.5786	10.3601	9.0400	1.2141	0.3975	0.0001	0.9884	0.0004	2332.5243	-44.4705	305.2269
25	7	2.1600	12.3984	10.2880	8.9800	1.2051	0.4033	0.0001	0.9880	0.0004	2348.2136	-44.8043	350.0312
26	8	2.1400	12.2194	10.2159	8.9200	1.1961	0.4092	0.0001	0.9875	0.0004	2364.9023	-45.1580	395.1892
27	9	2.1200	12.0416	10.1438	8.8600	1.1871	0.4152	0.0001	0.9871	0.0004	2382.6757	-45.5335	440.7227

VIGA 1 | VIGA 2 | PLACA 1 | predictor

Figura 3.17.
Solución Excel del método de Euler para la ecuación dinámica del flujo gradualmente variado

El paso siguiente es la formulación de un código computacional en MATLAB para la solución del caso con el método de Euler usando: $dy = \left(So \left[\frac{So - Se}{1 - \frac{Q^2 T}{gA^3}} \right] \right) dx$, $y_{i+1} = y_i + hf(i)$, $x_{i+1} = x_i + h$

```
%CURVA DE REMANSO METODO EULER
Q=5;
n=0.025;
S=0.0005;
z=1.5;
b=2.5;
Yn=1.375;
Yc=1.0;
dx=-4;
Yo=2.3;
for i=1:1000
if i==1
x(i)=0;
Y(i)=Yo;
Z(i)=0;
end
```

```
Sf(i)=(Q^2*n^2)/((b*Y(i)+z*Y(i)^2)^2*((b*Y(i)+z*Y(i)^2)/(
b+2*Y(i)*sqrt(1+z^2)))^(4/3));
F(i)=(S-Sf(i))/(1-((Q^2*(b+2*z*Y(i)))/(9.81*(b*Y(i)+z*Y(i)^2)^3)));
Y(i+1)=Y(i)+dx*F(i);
x(i+1)=x(i)+dx;
Z(i+1)=abs(S.*x(i));
end

[x',Y']
plot(x,Y+Z,x,Z+Yn,x,Z+Yc,x,Z)
xlabel('DISTANCIAS O PROGRESIVAS (m)')
ylabel('TIRANTES DE AGUA(m)')
title('CURVA DE REMANSO METODO EULER')
h = legend('tirantes calculados','tirante normal','tirante
critico','cota de fondo',1);
```

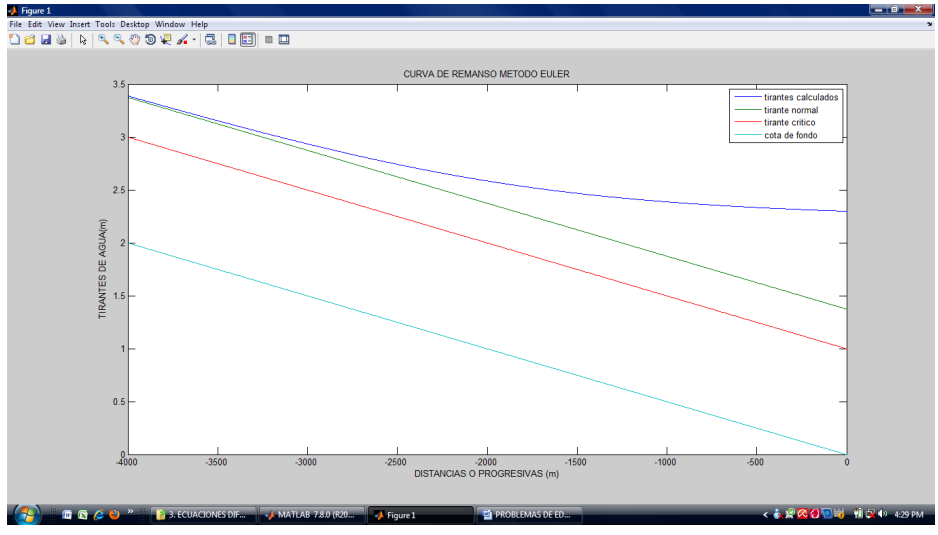


Figura 3.18.
Solución MATLAB del método de Euler para la ecuación dinámica del flujo gradualmente variado

También podemos emplear el método Runge-Kutta, el cual permite una mejor estructura del problema a resolver. La codificación en MATLAB es:

```
%CURVA DE REMANSO METODO RUNGE-KUTTA
Q=5;
n=0.025;
S=0.0005;
z=1.5;
b=2.5;
Yn=1.375;
Yc=1.0;
dx=-4;
```

```
Yo=2.3;
for i=1:1000
if i==1
x(i)=0;
Y(i)=Yo;
Z(i)=0;
End
Sf(i)=(Q^2*n^2)/((b*Y(i)+z*Y(i)^2)^2*((b*Y(i)+z*Y(i)^2)/
(b+2*Y(i)*sqrt(1+z^2)))^(4/3));
F(i)=(S-Sf(i))/(1-((Q^2*(b+2*z*Y(i)))/9.81*(b*Y(i)+z*Y(i)^2)^3)));
K1(i)=dx*F(i);
K2(i)=dx*(S-Sf(i))/(1-(Q^2*(b+2*z*(Y(i)+K1(i)/2)))/
(9.81*(b*(Y(i)+K1(i)/2)+z*(Y(i)+K1(i)/2)^2)^3)));
K3(i)=dx*(S-Sf(i))/(1-((Q^2*(b+2*z*(Y(i)+K2(i)/2)))/
(9.81*(b*(Y(i)+K2(i)/2)+z*(Y(i)+K2(i)/2)^2)^3)));
K4(i)=dx*(S-Sf(i))/(1-((Q^2*(b+2*z*(Y(i)+K3(i))))/
(9.81*(b*(Y(i)+K3(i))+z*(Y(i)+K3(i))^2)^3)));
Y(i+1)=Y(i)+(1/6)*(K1(i)+2*K2(i)+2*K3(i)+K4(i));
x(i+1)=x(i)+dx;
Z(i+1)=abs(S.*x(i));
end
[x',Y']
plot(x,Y+Z,x,Z+Yn,x,Z+Yc,x,Z)
xlabel('DISTANCIAS O PROGRESIVAS (m)')
ylabel('TIRANTES DE AGUA(m)')
title('CURVA DE REMANSO METODO RUNGE-KUTTA')
h = legend('tirantes calculados','tirante normal','tirante
critico','cota de fondo',1);
```

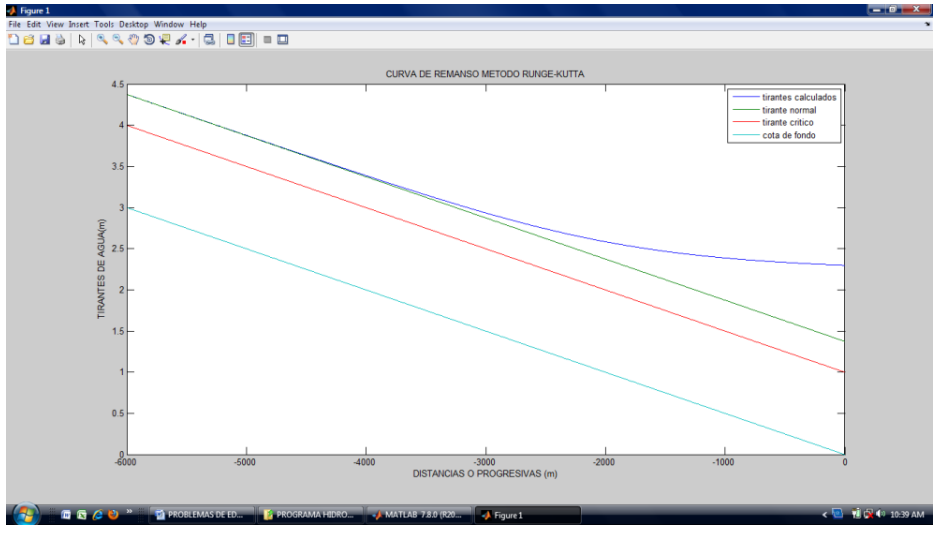


Figura 3.19.
Solución MATLAB del método Runge-Kutta para la ecuación dinámica del flujo gradualmente variado

3.7.4. Evolución de Temperatura en una Placa Calentada

Formulación del caso:

La ecuación diferencial mostrada representa el comportamiento de una placa calentada, la misma que se desea estudiar en cuanto a la evolución de su temperatura. Los valores propios para el material de la placa son:

ρ	300.00	Kg/m^3
V	0.001	m^3
A	0.250	m^2
C	900.000	$J/Kg°K$
hc	30.000	J/m^2K
ϵ	0.800	
σ	5.67E-08	$W/m^2°K^4$

$$\begin{cases} \dfrac{dT}{dt} = \dfrac{A}{\rho CV}\left[\varepsilon\sigma(297^4 - T^4) + h_c(297 - T)\right] \\ T(0) = 473 \end{cases} \tag{3.105}$$

Solución:

La solución, en primera instancia, la plantearemos desarrollando una formulación en hoja de cálculo Excel.

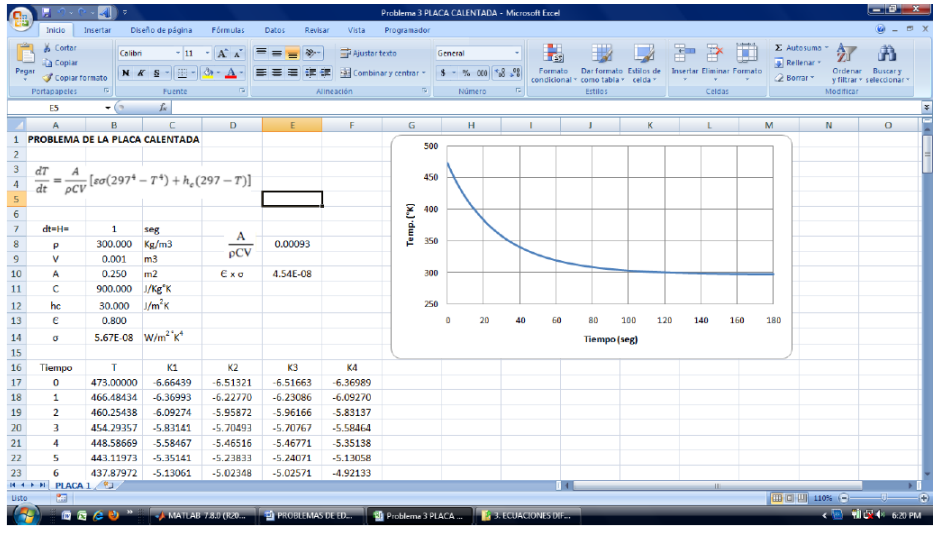


Figura 3.20.
Solución Excel para el problema de la placa calentada

Asimismo, proponemos un código computacional en MATLAB para la solución de este problema aplicando el método Runge-Kutta de orden 4.

```
%PLACA CALENTADA METODO RUNGE-KUTTA
clear, clc
format short
ro=300.000;
V=0.001;
A=0.25;
C=900;
hc=30;
e=0.8;
s=5.67E-08;
Q=A/(ro*C*V);
es=e*s;
dt=1;
Tf=180;
T(1)=473;
for i=1:Tf
K1(i)=dt*Q*(es*(294^4-(T(i))^4)+hc*(297-T(i)));
K2(i)=dt*Q*(es*(294^4-(0.5*K1(i)+T(i))^4)+hc*(297-(0.5*K1(i)+T(i))));
K3(i)=dt*Q*(es*(294^4-(0.5*K2(i)+T(i))^4)+hc*(297-(0.5*K2(i)+T(i))));
K4(i)=dt*Q*(es*(294^4-(K3(i)+T(i))^4)+hc*(297-(K3(i)+T(i))));
T(i+1)=T(i)+(1/6)*(K1(i)+2*K2(i)+2*K3(i)+K4(i));
y(i)=T(i);
end
w=1:Tf;
plot(w,y)
xlabel('TIEMPO TRANSCURRIDO (seg)')
ylabel('TEMPERATURAS(°K)')
title('CURVA DE TEMPERATURAS RUNGE-KUTTA')
h = legend('temperaturas calculadas',1);
```

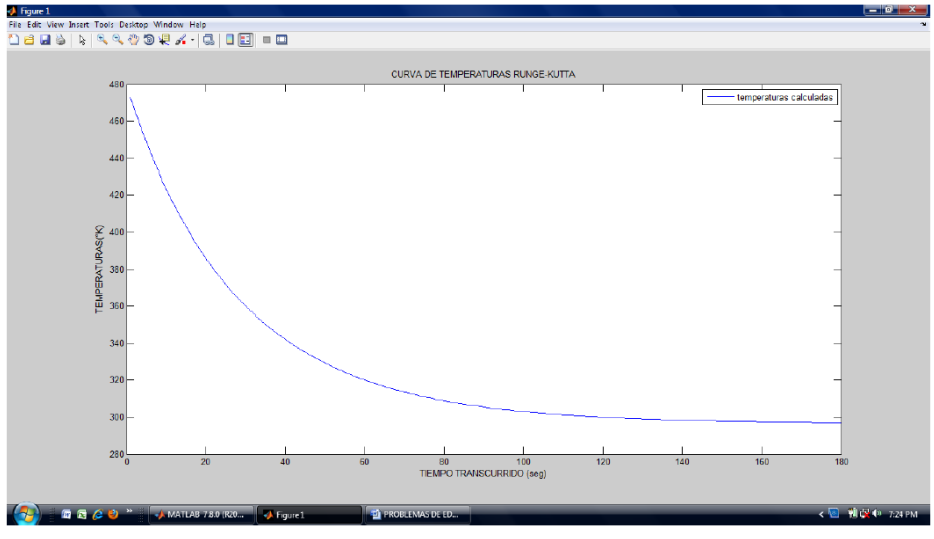


Figura 3.21.
Solución RK para el problema de la placa calentada

3.7.5. Modelo de Crecimiento Poblacional de Malthus

Formulación del caso:

Un modelo propuesto para el crecimiento poblacional alrededor del año 1800 es el propuesto por Malthus (1798), siendo el postulado el siguiente: "(…) la razón de cambio de la población (P) de bacterias en el instante (t) es proporcional a la población en el instante (…)". Las hipótesis de Malthus son:

- Las tasas de natalidad y mortalidad son constantes e independientes de la población o edad de individuos de la población.
- La reproducción se da de manera continua.
- El medio ambiente no influye.
- Es válida para una sola especie.

El modelo se rige por la siguiente ecuación diferencial en variable separable:

$$\frac{dP}{dt} = kP \tag{3.106}$$

$$P(0) = P_0 \tag{3.107}$$

Donde: k=a-b, "a" es la tasa de natalidad, "b" la tasa de mortalidad y P_0 la población inicial. Desarrollar una solución analítica y numérica para el estudio de la población utilizando los siguientes datos: constante k=0.1, Población inicial P_0=10 individuos. Encontrar la solución en el intervalo [0, 10] con 1000 intervalos equidistados.

Solución:

La resolución analítica que nos permite hallar la solución general del modelo es la siguiente:

$$\int \frac{dP}{P} = \int k dt \tag{3.108}$$
$$\ln\left(\frac{dP}{P}\right) = kt + c, \ldots\ldots$$
$$P(t) = ce^{kt}$$

Haciendo uso de las condiciones iniciales encontramos el valor del parámetro de integración c, usando en t=0, P=0.

$$P(0) = ce^{k0} = P_0 \tag{3.109}$$
$$c = P_0$$

Por lo tanto, la solución particular estará dada por:

$$P(t) = P_0 e^{kt} \tag{3.110}$$

En este caso realizaremos el estudio de la solución con cada uno de los métodos numéricos conocidos hasta ahora en términos de precisión y costo computacional. Para la solución del caso, estructuramos los archivos .m en MATLAB de la siguiente manera:

- Un archivo principal llamado **Poblacionm.m**
- Un archivo donde ingresamos la función llamada **Malthus.m**
- Ocho programas .m, denominados: eulerUV.m, eulermodUV.m, RunKutUV.m, AdBash4UV.m, AdBash5UV.m, AdBashMo4UV.m, AdBashMo5UV.m y MilneSimUV.m

En estos archivos se encuentra resuelto el algoritmo correspondiente a cada método, el cual con la función que se encuentra en Malthus.m se procesa en el programa principal denominado Poblacionm.m

De esta manera listamos los programas:

Programa: Malthus.m

```
function ypr=Malthus(x,y,z)
ypr=0.1*y;
```

Programa Principal: Poblacionm.m

```
%Programa Principal Para Modelo de crecimiento poblacional
de Malthus
clc
close all
clear all
%Datos iniciales
P0=10;
a=0;
b=10;
M=1000;
%Solucion analitica
h=(b-a)/M;
tt=a:h:b;
k=0.1;
solan=P0.*exp(0.1.*tt);
%Solucion Metodo de Euler
n11=cputime;
[y1,t1]=eulerUV('Malthus',a,b,P0,M);
n21=cputime;
tiem1=n21-n11;

%Solucion Metodo de Euler Modificado
n12=cputime;
```

```
[y2,t2]=eulermodUV('Malthus',a,b,P0,M);
n22=cputime;
tiem2=n22-n12;

%Solucion Metodo de Runge Kutta 4to Orden
n13=cputime;
[y3,t3]=RunKutUV('Malthus',a,b,P0,M);
n23=cputime;
tiem3=n23-n13;

%Solucion Metodo de Adams-Bashforth 4to Orden
n14=cputime;
[y4,t4]=AdBash4UV('Malthus',a,b,P0,M);
n24=cputime;
tiem4=n24-n14;

%Solucion Metodo de Adams-Bashforth 5to Orden
n15=cputime;
[y5,t5]=AdBash5UV('Malthus',a,b,P0,M);
n25=cputime;
tiem5=n25-n15;

%Solucion Metodo de Adams-Bashforth-Moulton 4to Orden
n16=cputime;
[y6,t6]=AdBashMo4UV('Malthus',a,b,P0,M);
n26=cputime;
tiem6=n26-n16;

%Solucion Metodo de Adams-Bashforth-Moulton 5to Orden
n17=cputime;
[y7,t7]=AdBashMo5UV('Malthus',a,b,P0,M);
n27=cputime;
tiem7=n27-n17;

%Solucion Metodo de Milne
n18=cputime;
[y8,t8]=MilneSim4UV('Malthus',a,b,P0,M);
n28=cputime;
tiem8=n28-n18;

%Graficador de soluciones de Malthus
subplot(2,2,1);
%h1=figure(1);

plot(t1,y1,t2,y2,t3,y3,t4,y4,t5,y5,t6,y6,t7,y7,t8,y8,tt,s
olan)
title('Modelo de crecimiento poblacional de Malthus: k=0,1,
h=0.01, Po=10')
xlabel('Tiempo (dias)')
```

```
ylabel('Individuos en la poblacion')
ylim([0 max(solan)+2])
xlim([a,b])
legend('Euler','Euler Modificado','Runge-Kutta 4','Adams-
Bashforth 4','Adams-Bashforth 5','Adams-Bashforth-Moulton
4','Adams-Bashforth-Moulton 5','Milne',2)
grid

%Calculo del error absoluto para las soluciones
E1=abs(y1-solan);
E2=abs(y2-solan);
E3=abs(y3-solan);
E4=abs(y4-solan);
E5=abs(y5-solan);
E6=abs(y6-solan);
E7=abs(y7-solan);
E8=abs(y8-solan);

% Grafica del error absoluto
subplot(2,2,4);
%h2=figure(2);
plot(t1,E1,t2,E2,t3,E3,t4,E4,t5,E5,t6,E6,t7,E7,t8,E8)
title('Errores del Modelo de crecimiento poblacional de
Malthus: k=0,1, h=0.01, Po=10')
xlabel('Tiempo (dias)')
ylabel('Error Absoluto')
xlim([a,b])
legend('Euler','Euler Modificado','Runge-Kutta 4','Adams-
Bashforth 4','Adams-Bashforth 5','Adams-Bashforth-Moulton
4','Adams-Bashforth-Moulton 5','Milne',2)
%legend('Error Euler','Error Euler Modificado','Error Runge-
Kutta 4','Error Adams-Bashforth 4','Error Adams-Bashforth
5','Error Adams-Bashforth-Moulton 4','Error Adams-
Bashforth-Moulton 5','Error Milne',2)
grid

%Calculo del error medio caudratico
Ecm1=sqrt(sum((y1-solan).^2/length(y1)));
Ecm2=sqrt(sum((y2-solan).^2/length(y2)));
Ecm3=sqrt(sum((y3-solan).^2/length(y3)));
Ecm4=sqrt(sum((y4-solan).^2/length(y4)));
Ecm5=sqrt(sum((y5-solan).^2/length(y5)));
Ecm6=sqrt(sum((y6-solan).^2/length(y6)));
Ecm7=sqrt(sum((y7-solan).^2/length(y7)));
Ecm8=sqrt(sum((y8-solan).^2/length(y8)));

ERMS=[Ecm1 Ecm2 Ecm3 Ecm4 Ecm5 Ecm6 Ecm7 Ecm8]
TiempoFinal=[tiem1 tiem2 tiem3 tiem4 tiem5 tiem6 tiem7
tiem8]
```

Programa eulerUV.m

```
function [y,x]=eulerUV(f,a,b,yi,M)
h=(b-a)/M;
x=a:h:b;
y(1)=yi;
for j=1:M
    y(j+1)=y(j)+h*feval(f,x(j),y(j));
end
```

Programa eulermodUV.m

```
function [y,x]=eulermodUV(f,a,b,yi,M)
h=(b-a)/M;
x=a:h:b;
y(1)=yi;
for j=1:M
    z=feval(f,x(j),y(j));
    k=feval(f,x(j+1),y(j)+h*z);
    y(j+1)=y(j)+h/2*(z+k);
end
```

Programa RunKutUV.m

```
function [y,x]=RunKutUV(f,a,b,yi,M)
h=(b-a)/M;
x=a:h:b;
y(1)=yi;
for j=1:M
    k1=h*feval(f,x(j),y(j));
    k2=h*feval(f,x(j)+h/2,y(j)+k1/2);
    k3=h*feval(f,x(j)+h/2,y(j)+k2/2);
    k4=h*feval(f,x(j)+h,y(j)+k3);
    y(j+1)=y(j)+(k1+2*k2+2*k3+k4)/6;
end
```

Programa AdBash4UV.m

```
function [y,x]=AdBash4UV(f,a,b,yi,M)
h=(b-a)/M;
x=a:h:b;
y(1)=yi;
%Aplica 4 pasos del metodo
for k=1:3
    k1=h*feval(f,x(k),y(k));
    k2=h*feval(f,x(k)+h/2,y(k)+k1/2);
    k3=h*feval(f,x(k)+h/2,y(k)+k2/2);
    k4=h*feval(f,x(k)+h,y(k)+k3);
    y(k+1)=y(k)+(k1+2*k2+2*k3+k4)/6;
```

```
end
%Inicia el metod predictor Adamas-Bashforth
for  k=4:M
     y(k+1)=y(k)+(h/24)*(55*feval(f,x(k),y(k))-
     59*feval(f,x(k-1),y(k-1))+37*feval(f,x(k-2),y(k-2))-
     9*feval(f,x(k-3),y(k-3)));
end
```

Programa AdBash5UV.m

```
function [y,x]=AdBash5UV(f,a,b,yi,M)
h=(b-a)/M;
x=a:h:b;
y(1)=yi;
%Aplica 4 pasos del metodo RK de 4to orden
for k=1:4
    k1=h*feval(f,x(k),y(k));
    k2=h*feval(f,x(k)+h/2,y(k)+k1/2);
    k3=h*feval(f,x(k)+h/2,y(k)+k2/2);
    k4=h*feval(f,x(k)+h,y(k)+k3);
    y(k+1)=y(k)+(k1+2*k2+2*k3+k4)/6;
end
%Inicia el metod predictor Adamas-Bashforth
for k=5:M
    y(k+1)=y(k)+(h/720)*(1901*feval(f,x(k),y(k))-
    2774*feval(f,x(k-1),y(k-1))+2616*feval(f,x(k-2),y(k-
    2))-1274*feval(f,x(k-3),y(k-3))+251*feval(f,x(k-
    4),y(k-4)));
end
```

Programa AdBashMo4UV.m

```
function [y,x]=AdBashMo4UV(f,a,b,yi,M)
h=(b-a)/M;
x=a:h:b;
y(1)=yi;
%Aplica 4 pasos del metodo
for k=1:3
    k1=h*feval(f,x(k),y(k));
    k2=h*feval(f,x(k)+h/2,y(k)+k1/2);
    k3=h*feval(f,x(k)+h/2,y(k)+k2/2);
    k4=h*feval(f,x(k)+h,y(k)+k3);
    y(k+1)=y(k)+(k1+2*k2+2*k3+k4)/6;
    z(k+1)=y(k+1);
end
%Inicia el metod predictor Adamas-Bashforth
for k=4:M
    z(k+1)=y(k)+(h/24)*(55*feval(f,x(k),y(k))-
    59*feval(f,x(k-1),y(k-1))+37*feval(f,x(k-2),y(k-2))-
```

```
    9*feval(f,x(k-3),y(k-3)));
    y(k+1)=y(k)+(h/24)*(9*feval(f,x(k+1),z(k+1))+19*
    feval(f,x(k),y(k))-5*feval(f,x(k-1),y(k-
    1))+1*feval(f,x(k-2),y(k-2)));
end
```

Programa AdBashMo5UV.m

```
function [y,x]=AdBashMo5UV(f,a,b,yi,M)
h=(b-a)/M;
x=a:h:b;
y(1)=yi;
%Aplica 4 pasos del metodo RK de 4to orden
for k=1:4
    k1=h*feval(f,x(k),y(k));
    k2=h*feval(f,x(k)+h/2,y(k)+k1/2);
    k3=h*feval(f,x(k)+h/2,y(k)+k2/2);
    k4=h*feval(f,x(k)+h,y(k)+k3);
    y(k+1)=y(k)+(k1+2*k2+2*k3+k4)/6;
    z(k+1)=y(k+1);
end
%Inicia el metod predictor Adamas-Bashforth
for k=5:M
    z(k+1)=y(k)+(h/720)*(1901*feval(f,x(k),y(k))-
    2774*feval(f,x(k-1),y(k-1))+2616*feval(f,x(k-2),y(k-2))-
    1274*feval(f,x(k-3),y(k-3))+251*feval(f,x(k-4),y(k-4)));
    y(k+1)=y(k)+(h/720)*(251*feval(f,x(k+1),z(k+1))+
    646*feval(f,x(k),y(k))-246*feval(f,x(k-1),y(k-
  1))+106*feval(f,x(k-2),y(k-2))-19*feval(f,x(k-3),y(k-3)));
end
```

Programa MilneSimUV.m

```
function [y,x]=MilneSim4UV(f,a,b,yi,M)
h=(b-a)/M;
x=a:h:b;
y(1)=yi;
%Aplica 4 pasos del metodo
for k=1:3
    k1=h*feval(f,x(k),y(k));
    k2=h*feval(f,x(k)+h/2,y(k)+k1/2);
    k3=h*feval(f,x(k)+h/2,y(k)+k2/2);
    k4=h*feval(f,x(k)+h,y(k)+k3);
    y(k+1)=y(k)+(k1+2*k2+2*k3+k4)/6;
    z(k+1)=y(k+1);
end
%Inicia el metod predictor Adamas-Bashforth
```

```
for k=4:M
    z(k+1)=y(k-3)+(4*h/3)*(2*feval(f,x(k),y(k))-
    feval(f,x(k-
    1),y(k-1))+2*feval(f,x(k-2),y(k-2))); y(k+1)=y(k-

1)+(h/3)*(feval(f,x(k+1),z(k+1))+4*feval(f,x(k),y(k))
+feval(f,x(k-1),y(k-1)));
end
```

Resultados:

Se muestran los resultados de error medio cuadrático y tiempo de cómputo de cada método según el orden que se muestra.

Euler, Euler Modificado, Runge-Kutta 4, Adams-Bashforth 4, Adams-Bashforth 5, Adams-Bashforth-Moulton 4, Adams-Bashforth-Moulton 5, Milne.

ERMS= [Ecm1 Ecm2 Ecm3 Ecm4 Ecm5 Ecm6 Ecm7 Ecm8]
[6.3178e-003 2.1062e-006 1.1554e-013 4.3746e-012 7.8794e-015 3.3613e-013 3.1711e-001 7.2162e-014]

TiempoFinal= [tiem1 tiem2 tiem3 tiem4 tiem5 tiem6 tiem7 tiem8]
[1.5600e-002 3.1200e-002 3.1200e-002 6.2400e-002 3.1200e-002 7.8001e-002 7.8000e-002 4.6800e-002]

En forma gráfica tenemos:

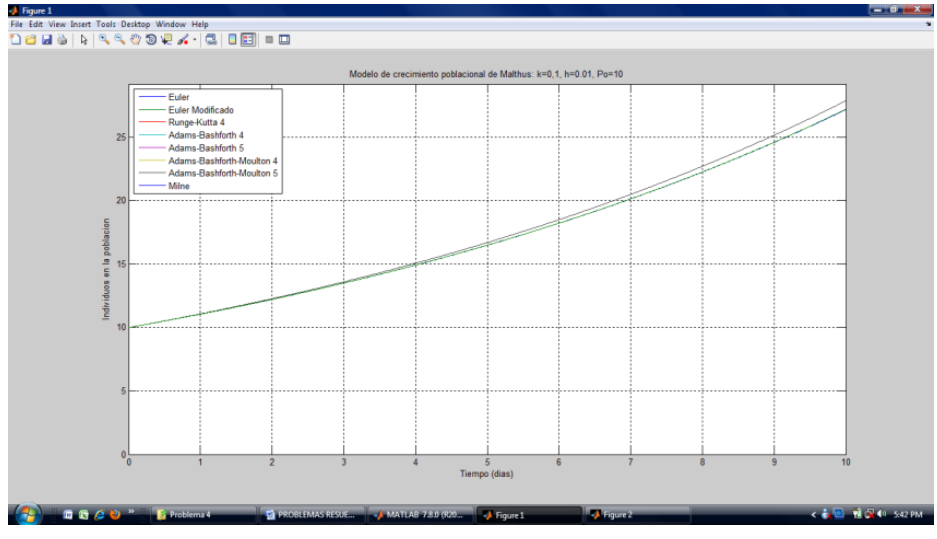
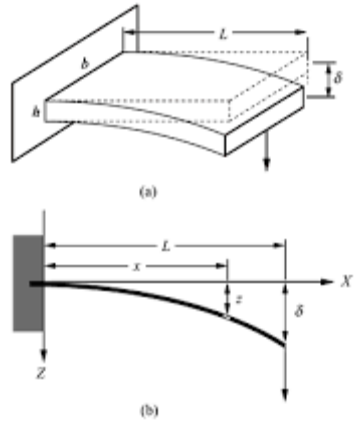


Figura 3.22.
Modelo de crecimiento poblacional de Malthus

3.7.6. Deflexión de una Viga en Voladizo

Formulación del caso:

Una viga de aluminio de 100 in de longitud, sección de 3 in de ancho y 0.8 in de alto, está empotrada en uno de sus extremos y es sometida en el otro a una carga P, hallar la deflexión y en el extremo libre, así como la curva elástica de la viga.



$$\frac{d^2y}{d^2x}=\frac{P}{EI}(L-x)\left[1+\left(\frac{dy}{dx}\right)^2\right]^{3/2} \tag{3.111}$$

Donde, P=carga aplicada en el extremo de la viga con una magnitud de 64 libras, E=módulo de elasticidad igual a 10^7 lb/in^2, I=momento de inercia.

Solución:

$$\frac{dy}{dx}=y'=z \tag{3.112}$$

$$\frac{d^2y}{d^2x}=y''=\frac{P}{EI}(L-x)[1+(z)^2]^{3/2} \tag{3.113}$$

Por lo tanto, tenemos:

$$F=z, \qquad G=\frac{P}{EI}(L-x)[1+(z)^2]^{3/2} \tag{3.114}$$

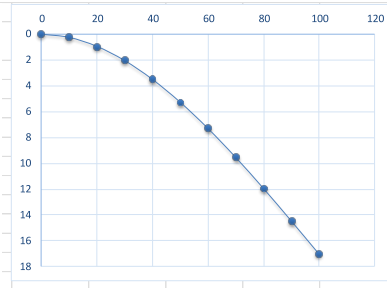
Como en todo método numérico buscamos un valor inicial o condición de frontera apoyado en condiciones físicas del problema, podemos decir que el extremo empotrado de la deflexión en los tres ejes es cero, por ello $x_0=y_0=z_0=0$. Además, debemos definir la longitud H de intervalo para el proceso de cálculo. La solución en Excel se muestra a continuación:

F	Z		
G	$=\frac{P}{EI}(L-x)[1+(z)^2]^{3/2}$		
H	10	pulg	
L	100	pulg	
E	1.00E+07	b	h
I	0.128	3	0.8
P	64		
K=P/EI	5.00E-05		

$$y_1 = y_o + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$z_1 = z_o + \frac{1}{6}(L_1 + 2L_2 + 2L_3 + L_4)$$

$$k_1 = hf(x_o, y_o, z_o)$$
$$L_1 = hg(x_o, y_o, z_o)$$
$$k_2 = hf\left(x_o + \frac{h}{2}, y_o + \frac{k_1}{2}, z_o + \frac{L_1}{2}\right)$$
$$L_2 = hg\left(x_o + \frac{h}{2}, y_o + \frac{k_1}{2}, z_o + \frac{L_1}{2}\right)$$
$$k_3 = hf\left(x_o + \frac{h}{2}, y_o + \frac{k_2}{2}, z_o + \frac{L_2}{2}\right)$$
$$L_3 = hg\left(x_o + \frac{h}{2}, y_o + \frac{k_2}{2}, z_o + \frac{L_2}{2}\right)$$
$$k_4 = hf(x_o + h, y_o + k_3, z_o + L_3)$$
$$L_4 = hg(x_o + h, y_o + k_3, z_o + L_3)$$



i	0	1	2	3	4	5	6	7	8	9	10
X	0	10	20	30	40	50	60	70	80	90	100
Y	0	0.2418	0.9353	2.0337	3.4908	5.2597	7.2922	9.5386	11.9473	14.4652	17.0380
Z	0	0.0476	0.0904	0.1285	0.1621	0.1909	0.2148	0.2336	0.2472	0.2554	
K1	0	0.4755	0.9037	1.2855	1.6209	1.9089	2.1479	2.3363	2.4723	2.5545	
L1	0.0500	0.0452	0.0405	0.0359	0.0312	0.0264	0.0214	0.0162	0.0109	0.0055	
K2	0.2500	0.7013	1.1061	1.4648	1.7768	2.0407	2.2549	2.4175	2.5269	2.5820	
L2	0.0475	0.0428	0.0382	0.0336	0.0288	0.0239	0.0189	0.0136	0.0082	0.0028	
K3	0.2377	0.6896	1.0946	1.4533	1.7649	2.0285	2.2422	2.4043	2.5134	2.5682	
L3	0.0475	0.0428	0.0382	0.0335	0.0288	0.0239	0.0188	0.0136	0.0082	0.0028	
K4	0.4754	0.9036	1.2854	1.6208	1.9088	2.1479	2.3363	2.4723	2.5545	2.5820	
L4	0.0452	0.0405	0.0359	0.0312	0.0264	0.0214	0.0162	0.0109	0.0055	0.0000	

Figura 3.23.
Solución Excel para la viga en voladizo

De la misma forma se elaboró un programa en MATLAB para la solución de la ecuación diferencial de la viga en voladizo usando el método Runge-Kutta.

```
%VIGA VOLADIZO METODO RUNGE-KUTTA
clear, clc
format short
P=64;
b=3;
h=0.8;
I=(b*h^3)/12;
L=100;
E=10^7;
R=P/(E*I);
H=1;
x(1)=0;
y(1)=0;
z(1)=0;
for i=1:H:L
K1(i)=H*z(i);
L1(i)=H*R*(L-x(i))*(1+(z(i))^2)^(3/2);
K2(i)=H*(z(i)+L1(i)/2);
L2(i)=H*R*(L-(x(i)+H/2))*(1+((z(i)+L1(i)/2)^2)^(3/2));
K3(i)=H*(z(i)+L2(i)/2);
L3(i)=H*R*(L-(x(i)+H/2))*(1+((z(i)+L2(i)/2)^2)^(3/2));
K4(i)=H*(z(i)+L3(i));
L4(i)=H*R*(L-(x(i)+H))*(1+((z(i)+L3(i))^2)^(3/2));
x(i+1)=x(i)+H;
y(i+1)=y(i)+(1/6)*(K1(i)+2*K2(i)+2*K3(i)+K4(i));
z(i+1)=z(i)+(1/6)*(L1(i)+2*L2(i)+2*L3(i)+L4(i));
x(i)=x(i+1);
y(i)=y(i+1);
```

```
z(i)=z(i+1);
def=y(i);
[i,def]
end
plot(x,y)
set(gca,'YDir','reverse')
grid
xlabel('Longitud de la Viga (pulg)')
ylabel('Deflexion(pulg)')
title('CURVA DE ELASTICA DE DEFLEXION METODO RUNGE-KUTTA')
h = legend('flechas calculadas',1);
```

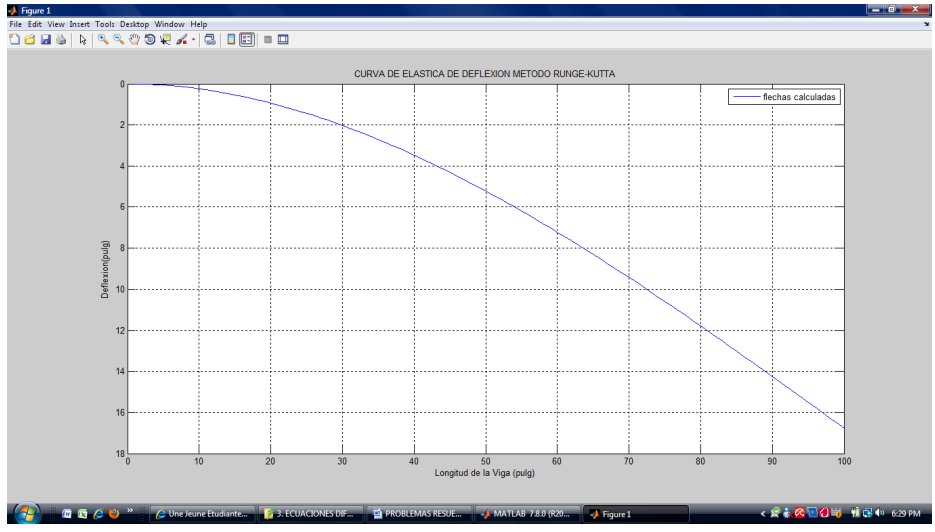
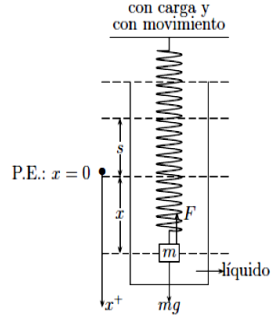


Figura 3.24.
Solución MATLAB para la viga en voladizo

3.7.7. Movimiento Amortiguado

Formulación del caso:

Cuando un cuerpo sujeto a un resorte se mueve en un medio que produce fricción sobre el cuerpo, entonces decimos que el movimiento se efectúa con amortiguamiento. Supongamos que el amortiguamiento es directamente proporcional a la velocidad, entonces por la segunda ley de Newton tenemos:



$$m\frac{d^2x}{dt^2} = -k(x+s) + mg - \beta\frac{dx}{dt} \tag{3.115}$$

Donde β es la constante de fricción o constante de amortiguamiento. Esta ecuación se puede escribir como:

$$\frac{d^2x}{dt^2} + 2\lambda\frac{dx}{dt} + \omega^2 x = 0 \tag{3.116}$$

Donde $2\lambda = \frac{\beta}{m} > 0$ y $\omega^2 = k/m$

Resolver la EDO para los siguientes casos y reproducir las funciones mostradas. Asumir las constantes necesarias para lograr los casos 1, 2 y 3.

- Caso 01: Cuando $\lambda - \omega^2 > 0$, estamos en un caso de un movimiento sobreamortiguado.
- Caso 02: Cuando $\lambda - \omega^2 = 0$, estamos en un caso de movimiento críticamente amortiguado.
- Caso 03: Cuando $\lambda - \omega^2 < 0$, estamos en caso de un movimiento subamortiguado.

Solución:

Para la solución del caso emplearemos el método de Runge-Kutta extendido.

Caso 1: Movimiento sobre amortiguado con h=0.2, λ=5, $\grave{\omega}$=3, $x(t)$=G, $x(0)$=3 y z=0. El coeficiente de amortiguamiento (β) es mayor respecto a la constante del resorte, por lo tanto, presenta un movimiento suave y no oscilatorio. La función $x(t)$ será:

$$x(t) = C_1 e^{P_1 t} + C_2 e^{P_2 t} \tag{3.117}$$

t	x	z	k1	l1	k2	l2	k3	l3	k4	l4
0	3	0	0	-5.4	-0.54	0	0	-4.914	-0.9828	4.428
0.2	2.6562	-1.8	-0.36	-1.18116	-0.478116	0.324	-0.3276	-1.0748556	-0.57497112	1.5582312
0.4	2.23179948	-1.98744	-0.397488	-0.04235906	-0.40172391	0.3577392	-0.36171408	-0.03854675	-0.40519735	0.68581978
0.6	1.84353926	-1.77379906	-0.35475981	0.22922746	-0.33183707	0.31928383	-0.32283143	0.20859699	-0.31304041	0.39313006
0.8	1.51401639	-1.49411254	-0.29882251	0.26299557	-0.27252295	0.26894026	-0.27192848	0.23932597	-0.25095731	0.2738149
1	1.24090261	-1.23522205	-0.24704441	0.2368194	-0.22336247	0.22233997	-0.22481041	0.21550566	-0.20394328	0.21046683
1.2	1.01634703	-1.0147258	-0.20294516	0.20002694	-0.18294247	0.18265064	-0.1846801	0.18202452	-0.16654026	0.16840208
1.4	0.83222528	-0.83176258	-0.16635252	0.16551966	-0.14980055	0.14971726	-0.15138079	0.15062289	-0.13622794	0.1367593
1.6	0.68140142	-0.68126937	-0.13625387	0.13601618	-0.12265226	0.12262849	-0.12399102	0.12377472	-0.11149893	0.11165058
1.8	0.55789486	-0.55785717	-0.11157143	0.1115036	-0.10042107	0.10041429	-0.10153001	0.10146827	-0.09127778	0.09132106
2	0.45676963	-0.45675888	-0.09135178	0.09133241	-0.08221853	0.0822166	-0.08313012	0.0831125	-0.07472928	0.07474163
2.2	0.37397324	-0.37397017	-0.07479403	0.07478851	-0.06731518	0.06731463	-0.06806257	0.06805754	-0.06118253	0.06118605
2.4	0.30618456	-0.30618369	-0.06123674	0.06123516	-0.05511322	0.05511306	-0.05572543	0.055724	-0.05009194	0.05009294
2.6	0.25068357	-0.25068332	-0.05013666	0.05013621	-0.04512304	0.045123	-0.04562436	0.04562395	-0.04101187	0.04101216
2.8	0.20524301	-0.20524294	-0.04104859	0.04104846	-0.03694374	0.03694373	-0.03735421	0.0373541	-0.03357777	0.03357785
3	0.1680393	-0.16803928	-0.03360786	0.03360782	-0.03024707	0.03024707	-0.03058315	0.03058311	-0.02749123	0.02749126
3.2	0.13757937	-0.13757937	-0.02751587	0.02751586	-0.02476429	0.02476429	-0.02503945	0.02503944	-0.02250799	0.02250799
3.4	0.11264082	-0.11264082	-0.02252816	0.02252816	-0.02027535	0.02027535	-0.02050063	0.02050063	-0.01842804	0.01842804
3.6	0.09222279	-0.09222279	-0.01844456	0.01844456	-0.0166001	0.0166001	-0.01678455	0.01678455	-0.01508765	0.01508765

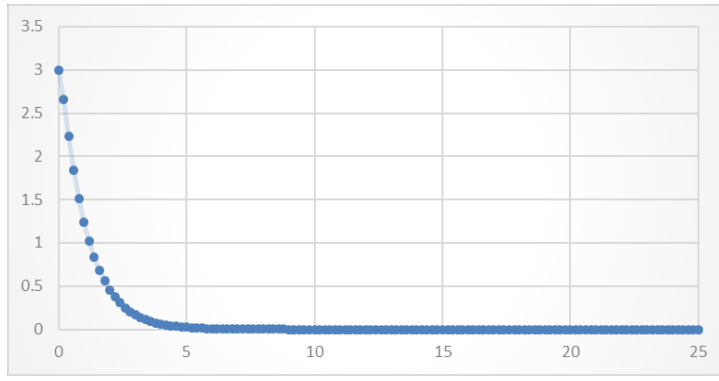


Figura 3.25.
Solución Excel del movimiento sobre amortiguado usando RK extendido

Caso 2: Movimiento críticamente amortiguado con h=1, λ=2, $\grave{\omega}$=2, $x(t)$=G, $x(0)$=1 y z=0. Estamos en un caso de movimiento críticamente amortiguado, cuando existe una pequeña disminución de la fuerza de amortiguamiento origina un movimiento oscilatorio. La función $x(t)$ será:

$$x(t) = C_1 e^{P_1 t} + C_2 e^{P_2 t} \tag{3.118}$$

t	x	z	k1	l1	k2	l2	k3	l3	k4	l4
0	1	0	0	-4	-2	4	2	-8	-8	20
1	-0.33333333	1.33333333	1.33333333	-4	-0.66666667	1.33333333	2	-5.33333333	-4	9.33333333
2	-0.33333333	0.88888889	0.88888889	-2.22222222	-0.22222222	0.44444444	1.11111111	-2.66666667	-1.77777778	4
3	-0.18518519	0.44444444	0.44444444	-1.03703704	-0.07407407	0.14814815	0.51851852	-1.18518519	-0.74074074	1.62962963
4	-0.08641975	0.19753086	0.19753086	-0.44444444	-0.02469136	0.04938272	0.22222222	-0.49382716	-0.2962963	0.64197531
5	-0.03703704	0.08230453	0.08230453	-0.18106996	-0.00823045	0.01646091	0.09053498	-0.19753086	-0.11522634	0.24691358
6	-0.01508916	0.03292181	0.03292181	-0.07133059	-0.00274348	0.00548697	0.03566529	-0.07681756	-0.04389575	0.09327846
7	-0.00594422	0.01280293	0.01280293	-0.02743484	-0.00091449	0.00182899	0.01371742	-0.02926383	-0.01646091	0.0347508
8	-0.00228624	0.00487731	0.00487731	-0.01036427	-0.00030483	0.00060966	0.00518214	-0.01097394	-0.00609663	0.01280293
9	-0.00086369	0.00182899	0.00182899	-0.0038612	-0.00010161	0.00020322	0.0019306	-0.00406442	-0.00223543	0.00467408
10	-0.00032177	0.0006774	0.0006774	-0.00142255	-3.387E-05	6.774E-05	0.00071127	-0.00149029	-0.00081288	0.00169351
11	-0.00011855	0.00024838	0.00024838	-0.00051934	-1.129E-05	2.258E-05	0.00025967	-0.00054192	-0.00029354	0.00060966
12	-4.3279E-05	9.032E-05	9.032E-05	-0.00018817	-3.7634E-06	7.5267E-06	9.4084E-05	-0.00019569	-0.00010537	0.00021827
13	-1.5681E-05	3.2616E-05	3.2616E-05	-6.774E-05	-1.2545E-06	2.5089E-06	3.387E-05	-7.0249E-05	-3.7634E-05	7.7776E-05
14	-5.645E-06	1.1708E-05	1.1708E-05	-2.4253E-05	-4.1815E-07	8.363E-07	1.2126E-05	-2.5089E-05	-1.3381E-05	2.7598E-05
15	-2.0211E-06	4.1815E-06	4.1815E-06	-8.6418E-06	-1.3938E-07	2.7877E-07	4.3209E-06	-8.9205E-06	-4.739E-06	9.7568E-06
16	-7.2015E-07	1.4868E-06	1.4868E-06	-3.0664E-06	-4.6461E-08	9.2922E-08	1.5332E-06	-3.1594E-06	-1.6726E-06	3.4381E-06
17	-2.5554E-07	5.2656E-07	5.2656E-07	-1.0841E-06	-1.5487E-08	3.0974E-08	5.4205E-07	-1.1151E-06	-5.8851E-07	1.208E-06
18	-9.0341E-08	1.8584E-07	1.8584E-07	-3.8201E-07	-5.1623E-09	1.0325E-08	1.9101E-07	-3.9234E-07	-2.0649E-07	4.2331E-07

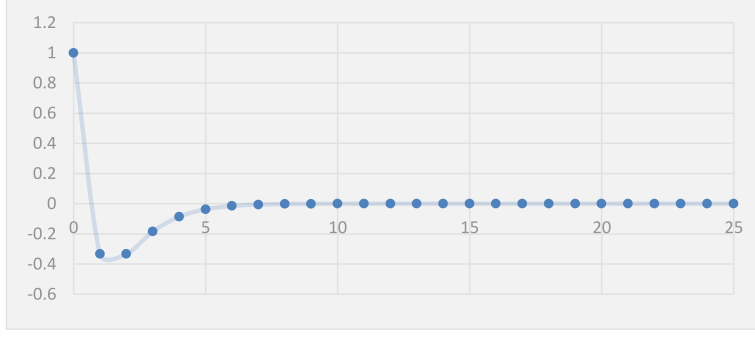


Figura 3.26.
Solución Excel del movimiento críticamente amortiguado usando RK extendido

Caso 3: Movimiento subamortiguado con h=0.5, λ=3, $\dot{\omega}$=5, $x(t)$=G, $x(0)$=1 y z=0. Estamos en un caso de movimiento críticamente amortiguado, el coeficiente de amortiguamiento es pequeño en comparación a la constante del resorte, lo cual genera raíces complejas en P1 y P2 por el coeficiente de $e^{-\lambda t}$, las amplitudes de vibración tienden a cero. La función $x(t)$ será:

$$x(t) = e^{-\lambda t}(C_1 cos\sqrt{\omega^2 - \lambda^2}\, t + C_2 sen\sqrt{\omega^2 - \lambda^2}\, t) \qquad (3.119)$$

t	x	z	k1	l1	k2	l2	k3	l3	k4	l4
0	1	0	0	-12.5	-3.125	6.25	1.5625	-2.34375	-1.171875	-25
0.5	0.28385417	-4.94791667	-2.47395833	11.2955729	0.3499349	9.81445313	-0.02034505	-5.61319987	-5.28055827	28.3894857
1	-0.89870199	3.06667752	1.53333876	2.03374227	2.04177433	-10.6002384	-1.11672084	5.1730103	4.11984391	0.47372182
1.5	0.35184629	1.67551217	0.83775609	-9.42461513	-1.5183977	-0.52366798	0.70683909	0.85087243	1.2631923	-20.8127211
2	0.43148482	-3.25497571	-1.62748785	4.37136689	-0.53464613	7.98611565	0.36904106	-4.26626825	-3.76062198	12.5571584
2.5	-0.52173518	0.80639431	0.40319715	4.10250681	1.42882386	-4.57123562	-0.73961175	2.02921113	1.41780272	7.26002028
3	0.01150217	1.85280733	0.92640366	-5.7021991	-0.49914611	-2.93892335	0.19167283	1.82584912	1.83932822	-13.5756568
3.5	0.36996639	-1.7311934	-0.8655967	0.56900033	-0.72334662	5.1254792	0.4157731	-2.59830212	-2.16474776	3.16674292
4	-0.23761553	-0.26617716	-0.13308858	3.76872556	0.80909281	-1.05255915	-0.39622837	0.29073423	0.01227853	7.84937748
4.5	-0.12012905	1.4162317	0.70811585	-2.74708194	0.02134537	-3.0521831	-0.05492992	1.69778417	1.55700794	-7.1538104
5	0.24619673	-0.68538333	-0.34269167	-1.02130908	-0.59801893	2.65247745	0.3204277	-1.26240691	-0.97389512	-1.23943456
5.5	-0.06576482	-0.59881709	-0.29940855	2.61851149	0.35521933	0.56204766	-0.15889663	-0.4446808	-0.52174894	5.93876177
6	-0.1371835	0.86651741	0.4332587	-0.88475847	0.21206909	-2.26548766	-0.13311321	1.18804124	1.02727932	-2.78496703
6.5	0.13255813	-0.10425232	-0.05212616	-1.34421965	-0.38818107	0.99789832	0.19734842	-0.41493543	-0.25959387	-2.56626862
7	0.01699391	-0.5616794	-0.2808397	1.47261438	0.08731389	1.01894094	-0.02610447	-0.60150887	-0.58159413	3.60344681
7.5	-0.10634192	0.42347482	0.21173741	0.05884959	0.22644981	-1.35278361	-0.12645849	0.6727137	0.54809426	-0.37856038
8	0.05362713	0.14349972	0.07174986	-1.10083825	-0.2034597	0.1019825	0.09724548	0.01781114	0.08065543	-2.36984023
8.5	0.04362327	-0.39501548	-0.19750774	0.63975556	-0.03756885	0.91454559	0.03112866	-0.49725752	-0.4461365	1.74241989
9	-0.0657975	0.14110979	0.07055489	0.3991393	0.17033974	-0.64053778	-0.08957955	0.29532268	0.21821623	0.63291574

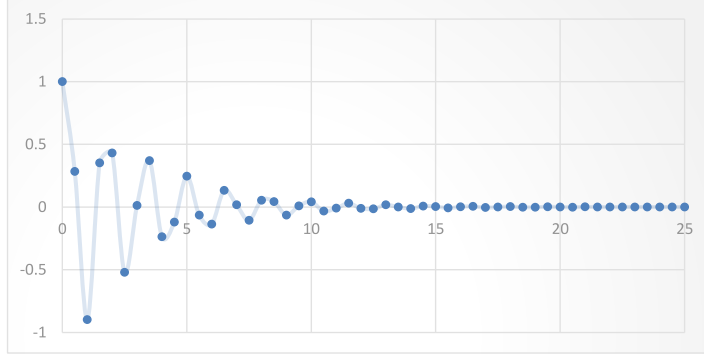


Figura 3.27.
Solución Excel del movimiento sub amortiguado usando RK extendido

La solución MATLAB se muestra a continuación:

ARCHIVO m

```
%MOVIMIENTO AMORTIGUADO
clear, clc
format short
W=input('ingrese la friccion del liquido: '); %2
G=input('ingrese el valor de landa : '); %5
L=input('ingrese el numero de interaciones: '); %100
H=input('ingrese el salto: '); %0.2
t=input('ingrese el tiempo inicial: '); %0
```

```
xin=input('ingrese el x inicial: '); %3
zin=input('ingrese el z inicial: '); %0
x(1)=t;
y(1)=xin;
z(1)=zin;
for i=1:1:L
    K1(i)=H*z(i);
    L1(i)=H*(-2*G* z(i)-W^2* y(i));
    K2(i)=H*(z(i)+L1(i)/2);
    L2(i)=H*(-2*G*(z(i)+L1(i)/2)-W^2* (y(i)+K1(i)/2));
    K3(i)=H*(z(i)+L2(i)/2);
    L3(i)=H*(-2*G*(z(i)+L2(i)/2)-W^2* (y(i)+K2(i)/2));
    K4(i)=H*(z(i)+L3(i));
    L4(i)=H*(-2*G*(z(i)+L3(i))-W^2* (y(i)+K3(i)));
    x(i+1)=x(i)+H;
    y(i+1)=y(i)+(1/6)*(K1(i)+2*K2(i)+2*K3(i)+K4(i));
    z(i+1)=z(i)+(1/6)*(L1(i)+2*L2(i)+2*L3(i)+L4(i));
    x(1)=t;
    y(1)=xin;
    z(1)=zin;
    x(i)=x(i+1);
    y(i)=y(i+1);
    z(i)=z(i+1);
    def=y(i);
    [i,def]
end
plot(x,y)
grid
xlabel('t')
ylabel('X(t)')
title('CURVA DE MOVIMIENTO AMORTIGUADO')
h = legend('flechas calculadas',10);
function pushbutton1_Callback(hObject, eventdata, handles)
W=str2double(get(handles.edit1,'string')); %2,2,5
G=str2double(get(handles.edit2,'string')); %5,2,3
L=str2double(get(handles.edit3,'string')); %100,25,50
H=str2double(get(handles.edit4,'string')); %0.2,1,0.5
to=str2double(get(handles.edit5,'string')); %0,0,0
xo=str2double(get(handles.edit6,'string')); %3,1,1
zo=str2double(get(handles.edit7,'string')); %0,0,0
for i=1:1:L
    x(1)=to;
    y(1)=xo;
    z(1)=zo;
    K1(i)=H*z(i);
    L1(i)=H*(-2*G* z(i)-W^2* y(i));
    K2(i)=H*(z(i)+L1(i)/2);
    L2(i)=H*(-2*G*(z(i)+L1(i)/2)-W^2* (y(i)+K1(i)/2));
    K3(i)=H*(z(i)+L2(i)/2);
```

```
    L3(i)=H*(-2*G*(z(i)+L2(i)/2)-W^2* (y(i)+K2(i)/2));
    K4(i)=H*(z(i)+L3(i));
    L4(i)=H*(-2*G*(z(i)+L3(i))-W^2* (y(i)+K3(i)));
    x(i+1)=x(i)+H;
    y(i+1)=y(i)+(1/6)*(K1(i)+2*K2(i)+2*K3(i)+K4(i));
    z(i+1)=z(i)+(1/6)*(L1(i)+2*L2(i)+2*L3(i)+L4(i));
    x(i)=x(i+1);
    y(i)=y(i+1);
    z(i)=z(i+1);
    MovAm=y(i);
    [i,MovAm];
end
set(handles.edit8,'string',MovAm);

ARCHIVO .fig

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
W=str2double(get(handles.edit1,'string')); %2,2,5
G=str2double(get(handles.edit2,'string')); %5,2,3
L=str2double(get(handles.edit3,'string')); %100,25,50
H=str2double(get(handles.edit4,'string')); %0.2,1,0.5
to=str2double(get(handles.edit5,'string')); %0,0,0
xo=str2double(get(handles.edit6,'string')); %3,1,1
zo=str2double(get(handles.edit7,'string')); %0,0,0
for i=1:1:L
    x(1)=to;
    y(1)=xo;
    z(1)=zo;
    K1(i)=H*z(i);
    L1(i)=H*(-2*G* z(i)-W^2* y(i));
    K2(i)=H*(z(i)+L1(i)/2);
    L2(i)=H*(-2*G*(z(i)+L1(i)/2)-W^2* (y(i)+K1(i)/2));
    K3(i)=H*(z(i)+L2(i)/2);
    L3(i)=H*(-2*G*(z(i)+L2(i)/2)-W^2* (y(i)+K2(i)/2));
    K4(i)=H*(z(i)+L3(i));
    L4(i)=H*(-2*G*(z(i)+L3(i))-W^2* (y(i)+K3(i)));
    x(i+1)=x(i)+H;
    y(i+1)=y(i)+(1/6)*(K1(i)+2*K2(i)+2*K3(i)+K4(i));
    z(i+1)=z(i)+(1/6)*(L1(i)+2*L2(i)+2*L3(i)+L4(i));
    x(i)=x(i+1);
    y(i)=y(i+1);
    z(i)=z(i+1);
    MovAm=y(i);
    [i,MovAm];
end
plot(x,y);
grid;
xlabel('t');
```

```
ylabel('X(t)');
title('CURVA DE MOVIMIENTO AMORTIGUADO');
h=legend('flechas calculadas');
```

Corrida del programa MATLAB; en el Apéndice B se dan lineamientos básicos para generar un programa en entorno GUIDE.

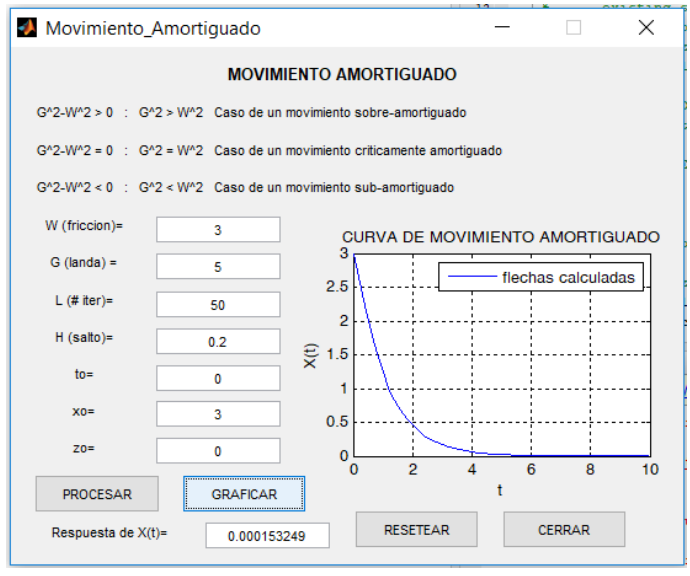


Figura 3.28.
Solución MATLAB del movimiento sobre amortiguado usando RK extendido

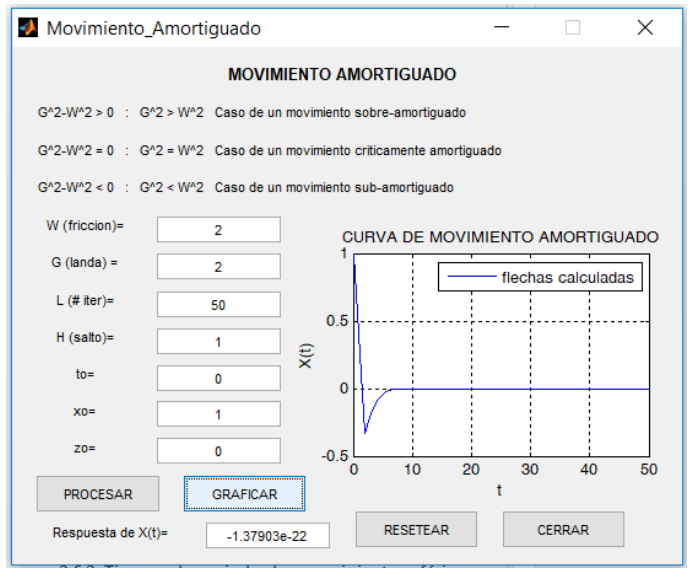


Figura 3.29.
Solución MATLAB del movimiento críticamente amortiguado usando RK extendido

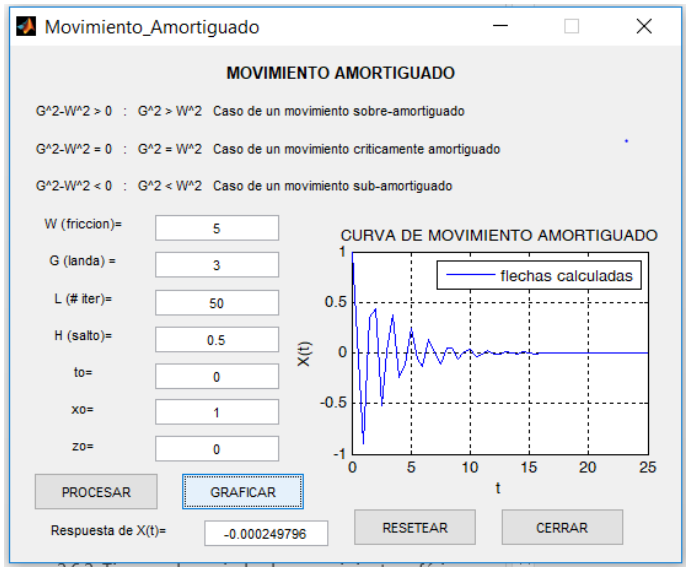


Figura 3.30.
Solución MATLAB del movimiento subamortiguado usando RK extendido

3.7.8. Velocidad de Caída de una Masa Esférica en Medio Acuoso

Formulación del caso:

La ecuación mostrada describe la velocidad terminal de un objeto esférico de masa m_s y volumen V relacionados en un fluido de densidad r; w se refiere a la velocidad vertical o de caída, g es la aceleración de la gravedad, C_D es el coeficiente de arrastre y A es el área de la sección transversal de la esfera.

$$m_s \frac{dw}{dt} = (m_s - rV)g - {}^1\!/_2\, rC_D A w^2 \qquad (3.120)$$

Para esferas pequeñas (R_D=wDr/m < 0.5), donde D es el diámetro de la esfera, m la viscosidad del fluido, C_D = 24/ R_D y la velocidad terminal w_s; es conocida como la ley de Stokes.

$$w_s = \frac{(r_s - r)gD^2}{18m} \qquad (3.121)$$

Donde r_s es la densidad de la esfera, realizar un cálculo usando D=3x10^{-5} m, r_s=2650 kg/m^3, r=1000 kg/m^3 y m=0.001 Pa-s.

Solución:

Se elaboró un programa stoks.m para *ode23* y *ode45*.

```
[t,w]=ode23('wstoks',[0 0.001],0);
[t2,w2]=ode45('wstoks',[0 0.001],0);
opt=odeset('RelTol',1e-6);   %reset tol value
```

```
[t3,w3]=ode23('wstoks',[0 0.001],0,opt);

%calcula Stokes
rho=1000; rhos=2650; mu=1.3e-3; g=9.81; D=.3e-4;
ws=(rhos-rho)*g*D.^2/(18*mu);
%check numero de Reynolds < 0.5 se requiere para la ley Stokes
Rd=ws*D*rho/mu;
if Rd>0.5, fprintf('Stokes eqn invalid'), end;
%check numero de Reynolds para la particula < 800 es requirido
if Rd>800, fprintf('Coeficiente de arrastre invalido'), end;

plot(t,w,'-b',t2,w2,'-r',t3,w3,'-g',max(t),ws,'*')
xlabel('t (s)'); ylabel('w (m/s)')
legend('ode23','ode45','ode23, rtol=1e-6','exacto')
```

Programa función wstoks.m

```
function wp=wstoks(t,w)

rho=1000; rhos=2650; mu=1.3e-3; g=9.81;
D=.3e-4; V=pi/6*D.^3; A=pi/4*D.^2; ms=rhos*V;
Rd=rho*D*w/mu; Cd=0;
if w~=0, Cd=24/Rd*(1+0.150*Rd^0.687); end;  %let Cd=0 if w=0
a=g*(ms-rho*V)/ms; b=0.5*rho*Cd*A ./ms;
wp=a-b*w.^2;
```

Los resultados se muestran en las siguientes curvas:

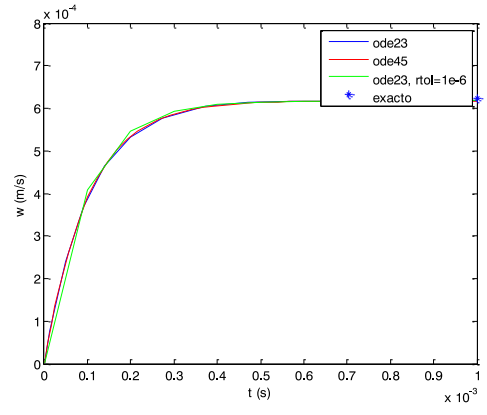


Figura 3.31.
Resultados MATLAB para la velocidad de caída de una masa esférica en medio acuoso

3.7.9. Modelo Lotka Volterra (Depredador-Presa)

Planteamiento del caso:

Una sola ecuación puede describir una población en un ambiente, pero si nos interesa estudiar dos poblaciones en un mismo medio, el modelo que utilizaremos debe tener en cuenta que ambas especies interactúan y

compiten en el mismo ambiente. El modelo demográfico de estas poblaciones $x(t)$ y $y(t)$ podría ser un sistema de dos ecuaciones diferenciales de primer orden escritas de la siguiente manera:

$$\frac{dx}{dt} = g_1(t,x,y) \qquad (3.122)$$

$$\frac{dy}{dt} = g_2(t,x,y)$$

A continuación, podemos diferenciar tres casos de este tipo de modelos, según la relación que existe entre las dos especies, modelo depredador-presa, modelo de competencia y modelo de cooperación entre especies.

Nos ocuparemos de resolver un modelo depredador-presa a través del método de la matriz exponencial (resolución de forma analítica), como del método numérico de Runge-Kutta (resolución de forma numérica), para poder comparar los resultados; sin embargo, a razón de que el método de la raíz exponencial solo puede aplicarse a sistemas de ecuaciones diferenciales lineales, el crecimiento poblacional no se comportará en forma proporcional a ambas poblaciones para resolverlo por este método exacto. Por lo tanto, usaremos el modelo depredador-presa lineal con coeficientes constantes a efecto de contar con la solución exacta.

En el caso de la especie presa $y(t)$, se supondrá que su población disminuye si la población de la especie depredadora $x(t)$ aumenta, a lo que se le llama efecto cruzado negativo, mientras que la especie depredadora incrementa su población cuando hay abundancia de presas. El sistema general de ecuaciones que gobierna el crecimiento relativo de las dos especies será:

$$\frac{dx}{dt} = g_1(t,x,y) = ax + by, \ldots\ldots\ldots x(t=0) = x_0 \qquad (3.123)$$

$$\frac{dy}{dt} = g_2(t,x,y) = -kx + hy, \ldots\ldots\ldots y(t=0) = y_0$$

Siendo todos los parámetros (a, b, h y k) positivos, el sistema de ecuaciones particular junto con los valores iniciales será:

$$\frac{dx}{dt} = 4x + 2y \qquad (3.124)$$

$$\frac{dy}{dt} = -x + y$$

Donde $x(t)$ representa la población de la especie depredadora y $y(t)$ la población de las presas, siendo las condiciones iniciales x_0=100 y y_0=400. La forma matricial será:

$$\begin{bmatrix} dx/dt \\ dy/dt \end{bmatrix} = \begin{bmatrix} 4 & 2 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{3.125}$$

La solución analítica del sistema planteado es:

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} -900\, e^{2t} + 1000\, e^{3t} \\ 900\, e^{2t} - 500\, e^{3t} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{3.126}$$

La solución numérica en hoja de cálculo la planteamos con Runge-Kutta de cuarto orden.

t	x	y	m1	k1	m2	k2	m3	k3	m4	k4
0.0000	100	400	30	7.5	31.6875	7.21875	31.7648438	7.19414063	33.5361914	6.88573242
0.0250	131.740146	407.201919	33.5341106	6.88654431	35.3829797	6.55344973	35.4670958	6.52617519	37.4071289	6.1630213
0.0500	167.180378	413.736722	37.4048739	6.16390858	39.4292153	5.77339651	39.5206696	5.74321085	41.6441014	5.31947211
0.0750	206.671836	419.489487	41.6416579	5.32044129	43.8567519	4.86642608	43.9561562	4.83306222	46.2789267	4.34236394
0.1000	250.596236	424.333118	46.2762795	4.34342205	48.698679	3.81926133	48.806695	3.78242932	51.3460704	3.2178154
0.1250	299.368419	428.127221	51.3432029	3.21897005	53.9908373	2.61741714	54.1081802	2.5768023	56.8828611	1.9306856
0.1500	353.439102	430.716903	56.8797554	1.93194503	59.7720418	1.2450974	59.8994849	1.20035823	62.9297218	0.46446686
0.1750	413.297857	431.931457	62.9263586	0.46584	66.0843225	-0.31491648	66.2227018	-0.36415049	69.5304212	-1.19883131
0.2000	479.476329	431.582936	69.5267797	-1.19733481	72.9731853	-2.08138624	73.1234043	-2.13551695	76.7323442	-3.07880784
0.2250	552.551712	429.464611	76.7284018	-3.07717752	80.4878925	-4.07474727	80.6509277	-4.13421052	84.5867841	-5.19680598
0.2500	633.150517	425.349295	84.5825164	-5.19503055	88.6817665	-6.31724988	88.8586735	-6.38251825	93.1492579	-7.57606034
0.2750	721.952626	418.987524	93.1446388	-7.57412755	97.6125175	-8.83311213	97.8044368	-8.90469792	102.479848	-10.2418559
0.3000	819.695692	410.10559	102.474849	-10.2397525	107.342597	-11.6486851	107.550761	-11.7271436	112.643568	-13.2217002
0.3250	927.179881	398.403405	112.638158	-13.2194119	117.939581	-14.7926315	118.165322	-14.8785645	123.710762	-16.545509
0.3500	1045.273	383.552186	123.704909	-16.5430204	129.476579	-18.2961195	129.721335	-18.3901791	135.757534	-20.2458083
0.3750	1174.91605	365.191948	135.751202	-20.2431025	142.032685	-22.1930313	142.298011	-22.2959239	148.866207	-24.3579508
0.4000	1317.12918	342.928788	148.859357	-24.3550098	155.69345	-26.5201894	155.981025	-26.6326803	163.125826	-28.9203525
0.4250	1473.0182	316.331937	163.118417	-28.9171566	170.551409	-31.3176013	170.863048	-31.4405193	178.632696	-33.9747458
0.4500	1643.78154	284.93058	178.624683	-33.971274	186.706635	-36.6287235	187.044297	-36.762966	195.490964	-39.5664556

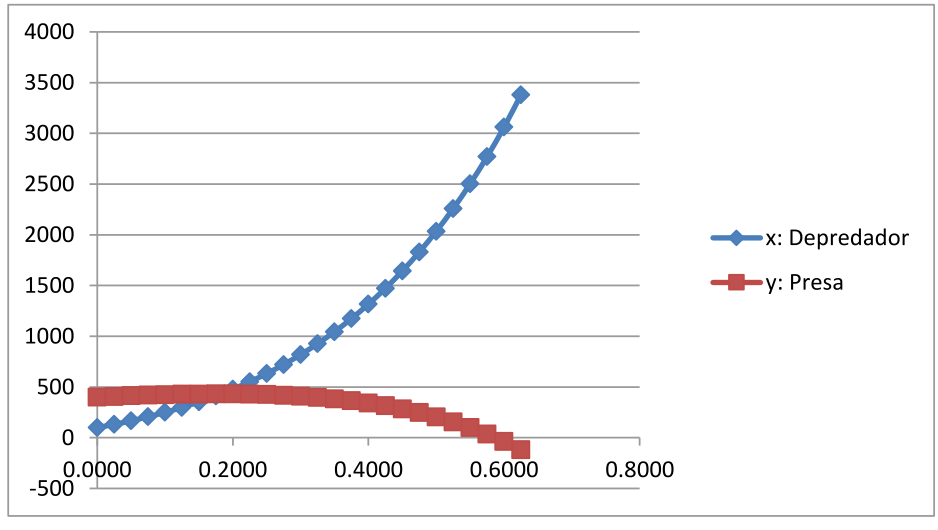


Figura 3.32.
Solución Excel para la evolución de la población Depredador-Presa

La solución con MATLAB posee gran versatilidad para tratar estos problemas de sistemas dinámicos por diversos medios. (a) Mediante el uso de matrices exponenciales. (b) Mediante el uso de comandos enlatados *ODE Solvers*. (c) Mediante la creación de un propio programa que implemente el algoritmo Runge-Kutta.

```
%Resolución de un sistema de ecuaciones método de la matriz
exponencial
clear, clc
A=[4 2; -1 1];  %Matriz de coeficiente
C=[100 400];    %condiciones iniciales
to=0;           %valor inicial del tiempo
tf=0.5;         %valor final del tiempo
h=0.025;        %longitud del paso
k=(tf-to)/h;    %calculo del numero de iteraciones
g=C;
tab=[to,C];
for  i=1:k
     to=to+h;
     w=(expm(A*(to))*C')';
     g=[g;w];
     tab2=[to,w];
     tab=[tab;tab2];
end
u=0:h:tf;
plot(u',g)
legend('Depredador','Presa');
tab
```

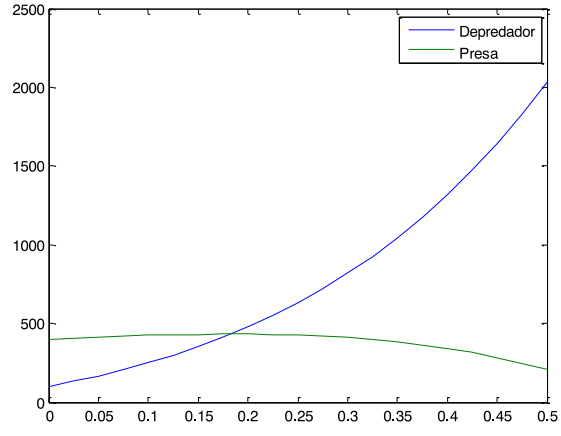


Figura 3.33.
Solución MATLAB para la evolución de la población Depredador-Presa

MATLAB cuenta con comandos predeterminados para resolver cualquier tipo de ecuaciones diferenciales como, por ejemplo, el *ode45*. A continuación, mostramos la creación de un sistema para utilizarlo en la representación gráfica.

```
function dy=dpl(t,y)
dy=[4*y(1)+2*y(2); -y(1)+y(2)];

[t,y]=ode45('dpl',[0 0.5],[100;400]);
plot(t,y(:,1),t,y(:,2))
xlabel('tiempo t');
ylabel('solucion y');
legend('y1:Depredador','y2:Presa');
sol=[t,y]
```

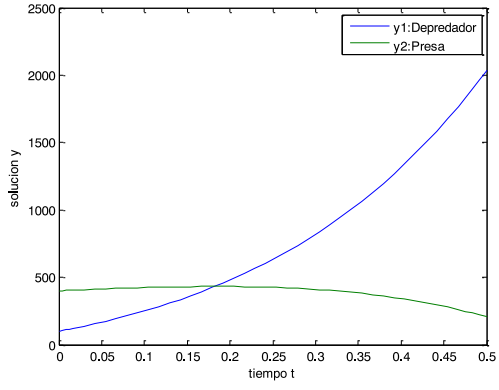
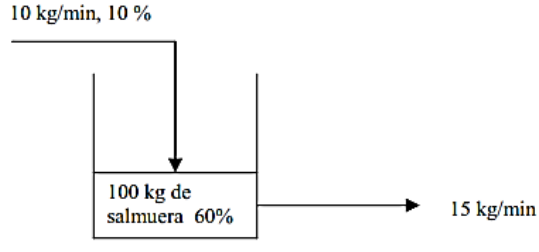


Figura 3.34.
Solución MATLAB ode45 para la evolución de la población Depredador-Presa

3.7.10. Balance de Masa en Estado No Estacionario

Planteamiento del caso:

Un tanque contiene 100 kg de una solución de salmuera al 60 % (60 % de sal) y se llena con una solución de sal al 10 % a una velocidad de 10 kg/min. La solución se saca del tanque a la velocidad de 15 kg/min. Suponiendo un mezclado completo, encuentre los kilogramos de sal en el tanque después de 10 minutos de iniciado el proceso.



Sea C los kilogramos de sal en el tanque al tiempo t; la ecuación de balance de masa en el tanque será:

$$\begin{bmatrix}\textit{Flujo de sal}\\ \textit{que entra}\end{bmatrix}-\begin{bmatrix}\textit{Flujo de sal}\\ \textit{que sale}\end{bmatrix}=\begin{bmatrix}\textit{Velocidad de}\\ \textit{acumulación}\\ \textit{de sal}\end{bmatrix} \tag{3.127}$$

$$(0.1)(10)-\frac{15C}{(100-(15-10)t)}=\frac{dC}{dt}$$

$$\frac{dC}{dt}+\frac{3C}{20-t}=1,\ldots\ldots\ldots\frac{dC}{dt}=1-\frac{3C}{20-t}$$

Solución:

Se elaboró una solución Excel para la ecuación de balance de masa en estado no estacionario practicando el uso de los métodos RK1, RK2, RK3, RK4, RK de orden superior. Los resultados se muestran como ejemplo para el método Runge-Kutta de primer orden.

Tiempo	C	K1	K2	K3	K4
0.000	60.00000	-2.00000	-1.97642	-1.97664	-1.95341
0.250	58.02334	-1.95342	-1.93013	-1.93035	-1.90741
0.500	56.09297	-1.90742	-1.88442	-1.88465	-1.86200
0.750	54.20830	-1.86201	-1.83931	-1.83953	-1.81718
1.000	52.36875	-1.81719	-1.79477	-1.79500	-1.77294
1.250	50.57373	-1.77295	-1.75083	-1.75105	-1.72929
1.500	48.82266	-1.72930	-1.70747	-1.70769	-1.68622
1.750	47.11494	-1.68623	-1.66470	-1.66492	-1.64374
2.000	45.45000	-1.64375	-1.62251	-1.62273	-1.60185
2.250	43.82725	-1.60186	-1.58091	-1.58113	-1.56054
2.500	42.24609	-1.56055	-1.53989	-1.54011	-1.51982
2.750	40.70596	-1.51982	-1.49946	-1.49968	-1.47968
3.000	39.20625	-1.47969	-1.45962	-1.45984	-1.44013
3.250	37.74639	-1.44014	-1.42036	-1.42058	-1.40116

9.500	12.48516	-0.64180	-0.62934	-0.62957	-0.61747
9.750	11.85557	-0.61748	-0.60532	-0.60555	-0.59373
10.000	11.25000	-0.59375	-0.58188	-0.58211	-0.57059

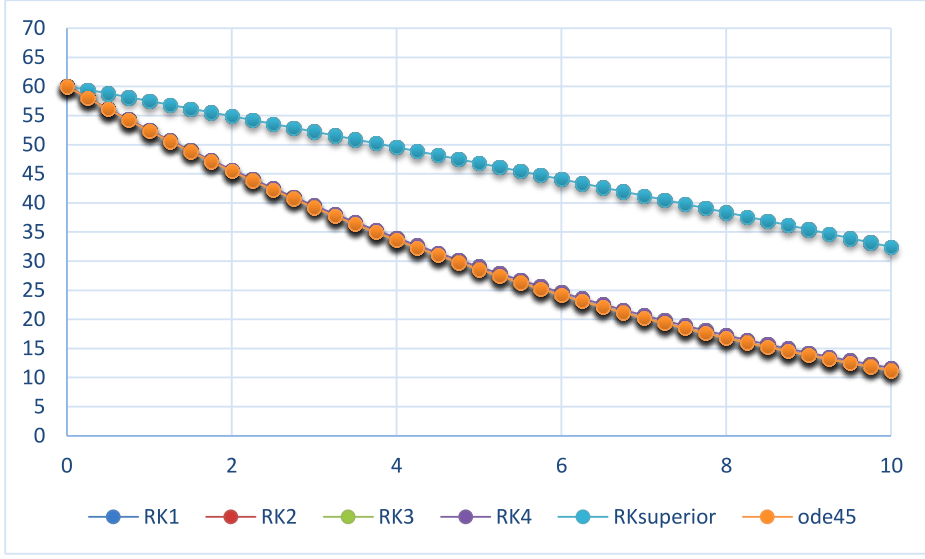


Figura 3.35.
Solución Excel para la ecuación de balance de masa en estado no estacionario

En forma gráfica se muestra la similitud de los resultados de los métodos RK, del primer al cuarto orden, presentándose una diferencia sustancial al usar el método RK de orden superior.

Asimismo, resolveremos la EDO usando MATLAB, siguiendo estos pasos:

- Determinar la condición inicial $C(0)=60$.
- Establecer el archivo "script" para función y archivo principal.
- Ejecución.

Para archivo función tendremos (def.m):

```
function dC=def(t,C)
dC=1-3*C/(20-t);
```

Archivo principal (def_ODE45)

```
[t,C]=ode45('def',[0:.25:10],60);
plot(t,C)
grid
disp([t,C])
```

Los resultados se muestran en la siguiente curva:

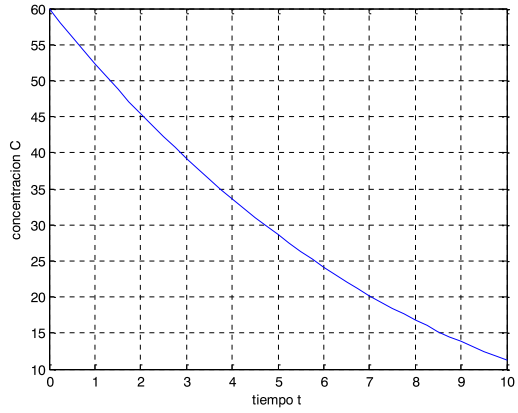


Figura 3.36.
Solución ode45 para la ecuación de balance de masa en estado no estacionario

3.7.11. Concentración de Contaminantes

Formulación del caso:

Un lago contaminado tiene una concentración inicial de una bacteria de 10^7 partes/m^3, mientras que el nivel aceptable es solo $5x10^6$. La concentración de la bacteria se reducirá a medida que el agua dulce ingrese al lago. Encuentra la concentración del contaminante después de 7 semanas.

La ecuación diferencial que gobierna la concentración C del contaminante en función del tiempo es:

$$\frac{dC}{dt} + 0.06\,C = 0, \ldots\ldots\ldots C(0) = 10^7 \qquad (3.128)$$

$$f(t,C) = -0.06\,C$$

La ecuación anterior presenta la siguiente solución analítica:

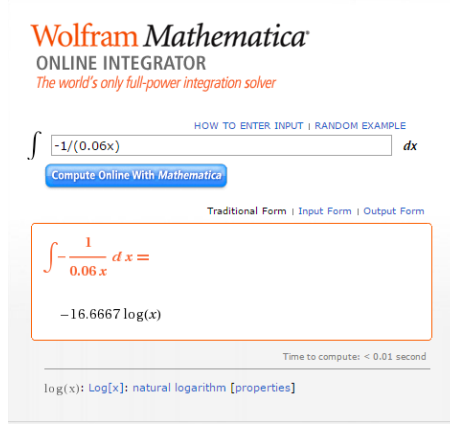


Figura 3.37.
Integración con Wolfram Mathematica, online integrator

La solución en Excel para Runge-Kutta de primer orden resulta:

Tiempo	C	K1	K2	K3	K4
0.00	10000000.00	-2100000.00	-1879500.00	-1891076.25	-1700442.98
3.50	8105875.34	-1702233.82	-1523499.27	-1532882.83	-1378357.88
7.00	6570521.50	-1379809.51	-1234929.52	-1242535.72	-1117279.71
10.50	5325982.82	-1118456.39	-1001018.47	-1007183.96	-905653.01
14.00	4317175.28	-906606.81	-811413.09	-816410.76	-734111.04
17.50	3499448.46	-734884.18	-657721.34	-661772.39	-595061.26
21.00	2836609.30	-595687.95	-533140.72	-536424.45	-482349.24
24.50	2299320.13	-482857.23	-432157.22	-434818.97	-390986.28
28.00	1863800.24	-391398.05	-350301.25	-352458.84	-316928.60
31.50	1510773.24	-317262.38	-283949.83	-285698.74	-256898.37
35.00	1224613.95	-257168.93	-230166.19	-231583.84	-208238.62
38.50	992656.80	-208457.93	-186569.85	-187718.97	-168795.63
42.00	804635.23	-168973.40	-151231.19	-152162.66	-136823.63

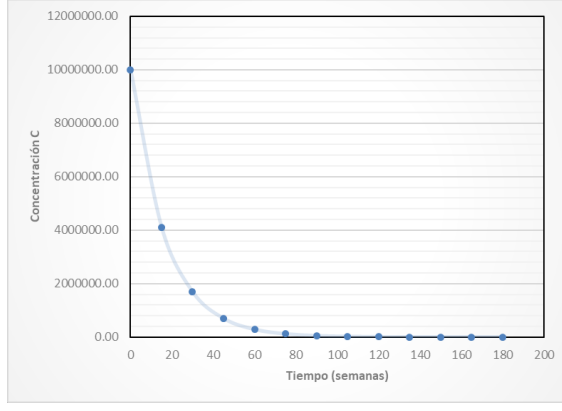


Figura 3.38.
Solución Excel para el decaimiento de la concentración del contaminante

A su vez se practicaron diferentes soluciones obteniéndose:

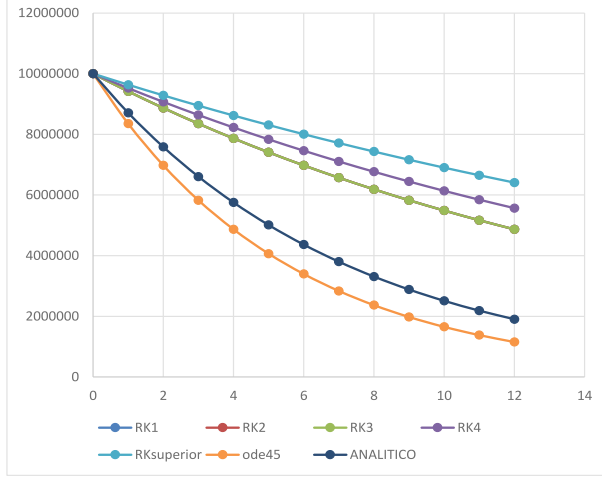


Figura 3.39.
Solución Excel para el decaimiento de la concentración del contaminante usando paso de tiempo h=1 semana

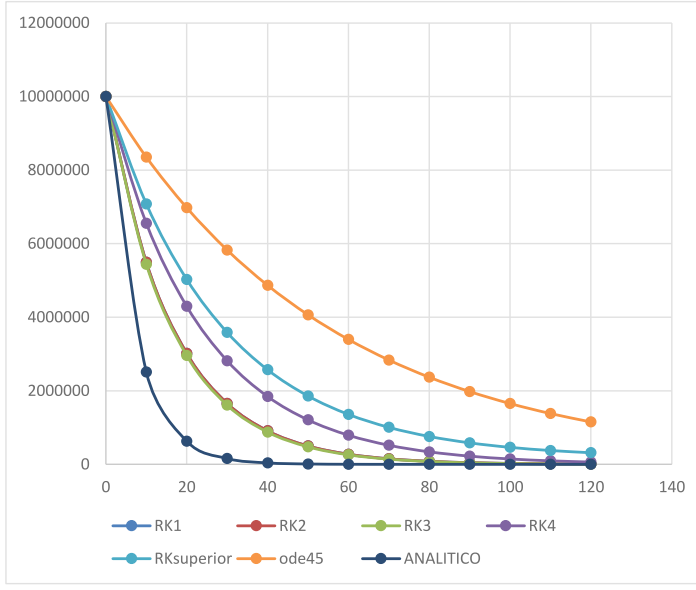


Figura 3.40.
Solución Excel para el decaimiento de la concentración del contaminante usando paso de tiempo h=10 semanas

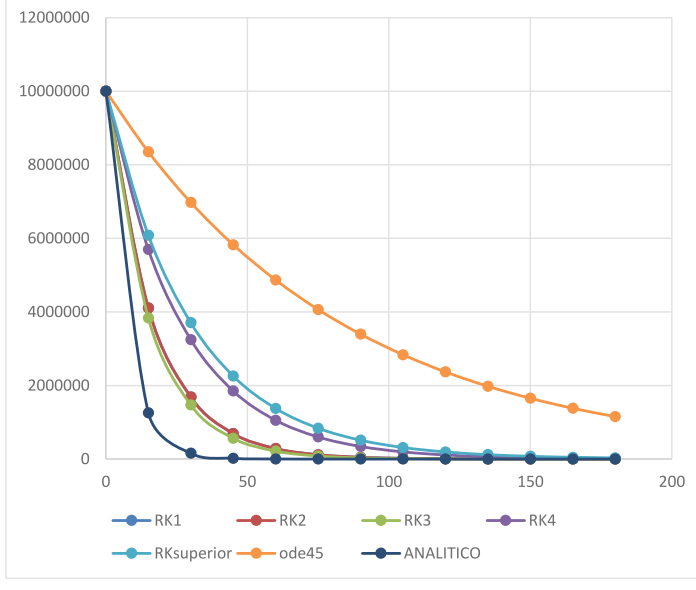


Figura 3.41.
Solución Excel para el decaimiento de la concentración del contaminante usando paso de tiempo h=15 semanas

Según estos resultados se puede concluir que existe una fuerte dependencia de la selección del paso de tiempo h. Asimismo, la solución *ode45* de MATLAB da resultados fuera del comportamiento normal de los demás métodos.

La solución *ode45* en MATLAB se estableció de la siguiente manera. Para el archivo función tendremos (def.m):

```
function dC=def(t,C)
dC=-0.06*C;
```

Archivo principal (def_ODE45)

```
[t,C]=ode45('def',[0:3:36],10E6);
plot(t,C)
xlabel('tiempo t');
ylabel('concentracion C');
grid
disp([t,C])
```

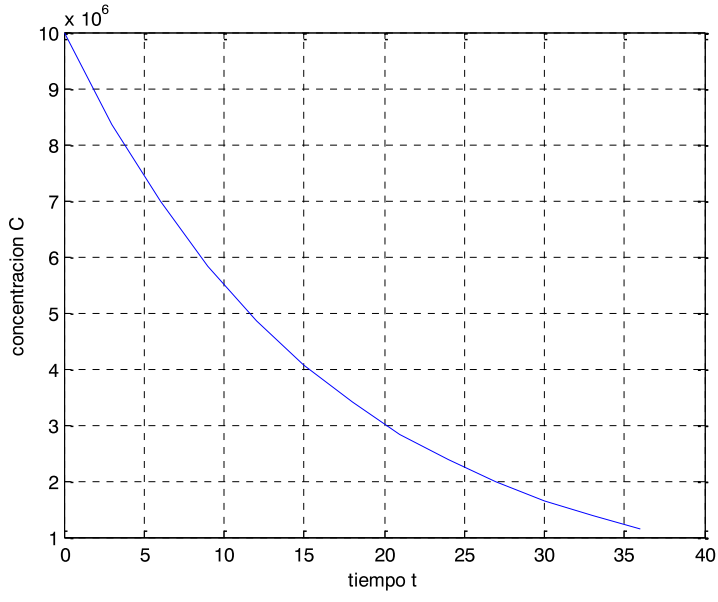


Figura 3.42.
Solución MATLAB para el decaimiento de la concentración del contaminante

3.8. Problemas Propuestos

- Problema propuesto 1: La rapidez de flujo de calor (conducción) entre dos puntos sobre un cilindro calentado en un extremo está dada por:

$$\frac{dQ}{dt} = \lambda A \frac{dT}{dx}$$

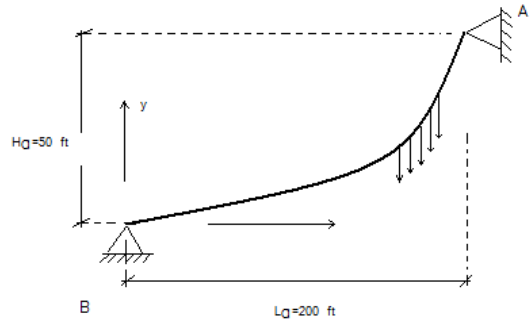
Donde "λ" es una constante, "A" es la sección transversal del cilindro, "Q" el flujo de calor, "T" la temperatura, "t" el tiempo y "x" la distancia desde el extremo calentado. Como la ecuación involucra dos variables, la simplificaremos haciendo:

$$\frac{dT}{dx} = \frac{100.(L-x)(20-t)}{100-xt}$$

Donde "L" es la longitud del cilindro. Combinar las dos ecuaciones y calcular el flujo de calor desde t=0 hasta 25 segundos. La condición inicial es $Q(0)$=0, y los parámetros λ=0.40 cal/cm-s, A=10 cm^2, L=20 cm y x=2.5 cm. Grafique sus resultados.

- Problema propuesto 2: Un cable cuelga de dos soportes: "A" y "B", dicho cable está con cargas distribuidas cuyas magnitudes varían con "x". como:

$$w = w_o\left[1 + sen\left(\frac{\pi x}{2l_a}\right)\right]$$



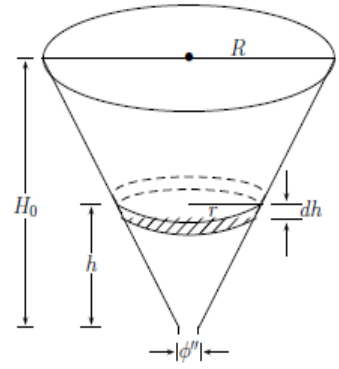
Donde w_o=1000 lb/ft, la pendiente del cable (dy/dx)=0 en x=0, que es el punto más bajo del cable. Este también es el punto del cable donde la tensión tiene un mínimo, T_o. La ecuación diferencial que modela el cable es:

$$\frac{d^2y}{dx^2} = \frac{w_o}{T_o}\left[1 + sen\left(\frac{\pi x}{2l_a}\right)\right]$$

Resuelva el caso usando un método numérico adecuado. Grafique la forma del cable (x, y).

- Problema propuesto 3: En la figura se muestra un cono circular de altura H_0 y radio R dispuesto verticalmente con orificio circular en el fondo de diámetro ϕ". Determinar el tiempo de vaciado en función de la altura (h). Si el diámetro del tubo de salida ϕ =2 in, R=1 m y H_0=1.2 m, el sistema generado para la EDO y los valores iniciales es:

$$\begin{cases} h^{3/2}\frac{dh}{dt} = -\frac{4{,}8\phi^2 H_0^5}{576R^2} \\ h(t=0) = Ho = 1{,}2 \\ h(t=?) = 0 \end{cases}$$



- Problema propuesto 4: En un tanque perfectamente agitado se tiene 400 litros de una salmuera en la cual están disueltos 25 Kg de sal común (ClNa), en cierto momento se incorpora al tanque un gasto de 80 l/min de una salmuera que contiene 0.5 Kg de sal común por litro. Si se tiene un gasto de salida de 80 l/min, determine:

(a) Qué cantidad de sal hay en el tanque, transcurridos 10 minutos.
(b) Qué cantidad de sal hay en el tanque, transcurrido un tiempo muy grande.

Si llamamos "x" a los kg de sal en el tanque después de "t" minutos, la acumulación de sal en el tanque estará dada por:

$$\frac{dx}{dt} = masa\ de\ sal\ q'entra - masa\ de\ sal\ q'sale$$

$$\frac{dx}{dt} = 80(0{,}5) - 80\left(\frac{x}{400}\right) = 40 - 0{,}2x$$

Usando la condición inicial de que hay 25 kg de sal en el tiempo cero tenemos:

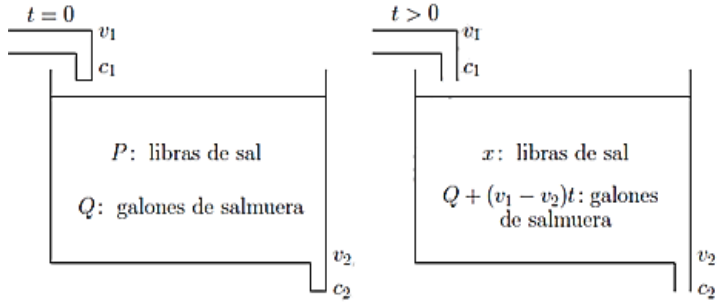
$$\begin{cases} \frac{dx}{dt} = 40 - 0{,}2x \\ x(t=0) = 25 \\ x(t=10) = ¿ \end{cases}$$

- Problema propuesto 5: Una salmuera (solución de sal en agua) entra en un tanque a una velocidad v_1=5 galones de salmuera/minuto y con una concentración de c_1=1.5 libras de sal por galón de salmuera (lib. sal/gal. salmuera). Inicialmente el tanque tiene Q=50 galones de salmuera con P=10 libras de sal disueltas. La mezcla bien homogenizada sale del tanque a una velocidad de v_2=8 galones de salmuera/minuto. La Ecuación diferencial que gobierna este proceso es:

$$\frac{dx}{dt} = v_1\left(gal.\frac{sol.}{\min.}\right)c_1\left(\frac{lib.sal}{gal.sol}\right) - v_2\left(gal.\frac{sol.}{\min.}\right)c_2\left(\frac{lib.sal}{gal.sol}\right)$$

$$\frac{dx}{dt} = v_1c_1 - v_2\frac{x}{Q+(v_1-v_2)t}$$

$$\frac{dx}{dt} + \frac{v_2}{Q+(v_1-v_2)t}x = v_1c_1 = q(t)$$



Esta última representa una EDO lineal de primer orden en "x". Resolver la EDO para un intervalo de tiempo muy largo.

- Problema propuesto 6: Sea "x" la temperatura al tiempo "t" de un cuerpo inmerso en un medio cuya temperatura está descrita por la expresión:

$$G(t) = -100(t-4)$$

La temperatura satisface la siguiente ecuación diferencial:

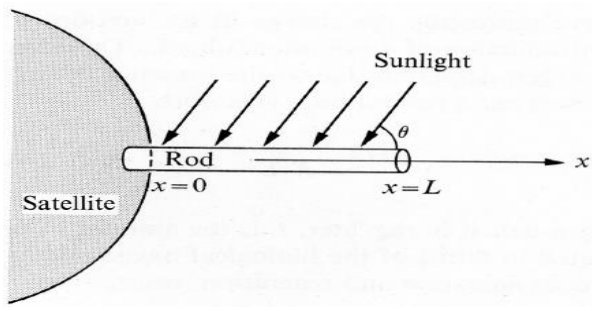
$$\frac{dx}{dt} + kx = kG(t)$$

Donde k=0.045 es una constante de proporcionalidad y la condición inicial es que en t=0, x=1000 °F. Obtener la relación x-t con el procedimiento de Runge-Kutta y comparar el resultado con la solución analítica que se muestra a continuación.

$$x = 400 - \left(\frac{100}{k}\right)(kt-1) + \left(600 - \frac{100}{k}\right)\exp(-kt)$$

- Problema propuesto 7: Una barra sobresale de un satélite en un campo de radiación solar. La ecuación diferencial basada en un modelo unidimensional de conducción de calor es:

$$\frac{dT}{dx} = \sqrt{\frac{2\sigma\epsilon CT^4}{5kA} - \frac{2S\alpha DT\sin\theta}{kA}} + K$$



Donde:
α = Absortividad = 0.4
ϵ = Emisividad = 0.4
A = Área transversal de la barra
C = Circunferencia de la barra
D = Diámetro de la barra = 1 in
K = Constante de integración
L = Longitud de la barra = 3 ft
S = Constante de la radiación solar = 425 BTU/h. ft^2
T = Temperatura de la barra en °R
k = Conductividad térmica de la barra = 100 BTU/h.ft.°R
σ = Constante de Stefan-Boltzmann = 0.173 x 10^{-8} BTU/h.ft2.°R4
θ = 30°

La condición inicial es que en x=3.0 ft, T=637.6285 °R, y dT/dx=0. Encontrar la relación $T(x)$.

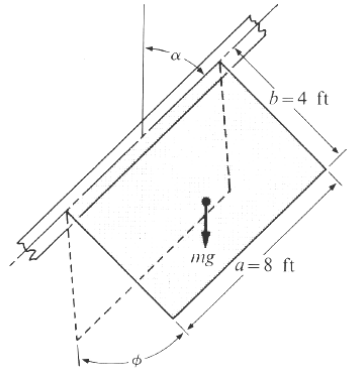
- Problema propuesto 8: La ecuación diferencial que describe una esfera en la que hay una fuente de calor en su centro se expresa como:

$$\frac{d^2T}{dr^2} + \left(\frac{2}{r}\right)\frac{dT}{dr} + \frac{w}{k} = 0$$

Donde T es la temperatura dentro de la esfera y k es la conductividad térmica de la esfera en BTU/h.ft °F. La fuente de calor se mantiene a una temperatura constante Tc, mientras que el gradiente de temperatura dT/dr en el centro es cero. El diámetro de la esfera es igual a 2 ft. Los demás valores son: w=100 BTU/h.ft^3, k=212 BTU/h.ft °F y Tc=900 °F. ¿Cuál es la distribución radial de temperatura de la esfera?

- Problema propuesto 9: Una puerta cuelga de un soporte, sin fricción, inclinado, como muestra la figura. La ecuación diferencial que expresa la oscilación de la puerta es:

$$\frac{d^2\phi}{dt^2} + \frac{3g}{2b}\sin\propto\sin\phi = 0$$



La aceleración de la gravedad es g=32.2 ft/s^2, sean los valores iniciales ϕ_0 = 90°, $(d\phi/dt)_{t=0}$ =4π rad/s. Resolver la ecuación en el intervalo 0 < t < 3 s.

CAPÍTULO IV
ECUACIONES DIFERENCIALES PARCIALES (EDP)

4.1. Generalidades

A modo de introducción a la resolución numérica de ecuaciones diferenciales en derivadas parciales, recordemos algunos conceptos básicos:

- Se denomina ecuaciones diferenciales parciales (EDP) a aquellas ecuaciones que involucran derivadas parciales de una función desconocida con dos o más variables independientes.
- Se denomina orden de una ecuación diferencial al orden de la derivada más alta que exista en dicha ecuación.
- Una ecuación diferencial parcial lineal es aquella que es lineal en la función desconocida y en todas sus derivadas, con coeficientes que dependen solo de las variables independientes de la función.

Veamos a continuación ejemplos de ecuaciones diferenciales en derivadas parciales:

$$\frac{\partial^2 u}{\partial x^2} + 2xy\frac{\partial^2 u}{\partial y^2} + u = 1 \ldots\ldots \textit{lineal de segundo orden} \tag{4.129}$$

$$\frac{\partial^3 u}{\partial x^2 \partial y} + x\frac{\partial^2 u}{\partial y^2} + 8u = 5y \ldots\ldots \textit{lineal de tercer orden}$$

$$\left(\frac{\partial^2 u}{\partial x^2}\right)^x + 2\frac{\partial^3 u}{\partial x \partial y^2} = x \ldots\ldots \textit{no lineal de tercer orden}$$

$$\frac{\partial^2 u}{\partial x^2} + xu\frac{\partial u}{\partial y} = x \ldots\ldots \textit{no lineal de segundo orden}$$

La mayoría de los problemas de importancia práctica en ciencias e ingeniería están descritos por este tipo de ecuaciones diferenciales y, fundamentalmente, por ecuaciones diferenciales de segundo orden. Por ello el tratamiento de las EDP que se desarrollará en lo sucesivo se concentrará en ecuaciones lineales de segundo orden.

En este capítulo solo se va a utilizar el método de diferencias finitas en la resolución de problemas; este método comúnmente se aplica cuando los dominios son rectangulares, poligonales paralelos a los ejes o circulares. Para dominios más elaborados debe usarse el método de elementos finitos, que abordaremos en el Capítulo V, específicamente usando, PDETool de MATLAB.

4.2. Clasificación Matemática

Las ecuaciones diferenciales de segundo orden en derivadas parciales pueden expresarse de forma general como:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \tag{4.130}$$

$$A\frac{\partial^2 u}{\partial x^2} + B\frac{\partial^2 u}{\partial x \partial y} + C\frac{\partial^2 u}{\partial y^2} + D = 0$$

Donde A, B y C son funciones de x y de y, y D es una función de x, y, u, $\partial u/\partial x$ y $\partial u/\partial y$. Es decir que estamos asumiendo que esta ecuación es lineal. Dependiendo de los valores de los coeficientes de los términos de la segunda derivada A, B y C la ecuación anterior puede clasificarse en una de las tres categorías siguientes (Chapra & Canale, 2005):

$B^2 - 4AC$	Categoría	Ejemplo
< 0	Elíptica	Ecuación de Laplace (estado estacionario con dos dimensiones espaciales) $\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$
= 0	Parabólica	Ecuación de conducción del calor (variable de tiempo y una dimensión espacial) $\frac{\partial T}{\partial t} = k' \frac{\partial^2 T}{\partial x^2}$
> 0	Hiperbólica	Ecuación de onda (variable de tiempo y una dimensión espacial) $\frac{\partial^2 y}{\partial x^2} = \frac{1}{c^2}\frac{\partial^2 y}{\partial t^2}$

Tabla 4.1.
Clasificación de las EDP

Esta clasificación es útil por dos razones:

- Cada grupo está asociado a diferentes problemas específicos en ciencias e ingeniería.
- Cada grupo requiere técnicas de solución especiales.

La terminología utilizada para clasificar a las ecuaciones surge por analogía con la utilizada en la clasificación de ecuaciones generales de segundo orden en la geometría analítica. Es importante notar que para los casos donde A, B y C dependen de x y de y, la ecuación puede estar en una categoría diferente, dependiendo del dominio para el cual se quiere calcular dicha ecuación.

4.3. Ecuaciones Elípticas

Este tipo de ecuaciones permite resolver los llamados problemas de equilibrio, que son problemas donde se busca la solución de una ecuación diferencial dada, en un dominio cerrado, sujeta a condiciones de frontera prescriptas; es decir que los problemas de equilibrio son problemas de condiciones de frontera. Los ejemplos más comunes de tales problemas incluyen distribuciones estacionarias de temperatura, flujo de fluidos incompresibles no viscosos, distribución de tensiones en sólidos en equilibrio, el campo eléctrico en una región que contenga una densidad de carga dada y, en general, problemas donde el objetivo sea determinar un potencial.

4.4. Ecuaciones Parabólicas

Este tipo de ecuaciones permite resolver los denominados problemas de propagación, que son problemas transitorios donde la solución de la ecuación diferencial parcial es requerida sobre un dominio abierto, sujeta a condiciones iniciales y de frontera prescritas. Los ejemplos más comunes de estos problemas incluyen a problemas de conducción de calor, problemas de difusión y, en general, problemas donde la solución cambia con el tiempo.

4.5. Ecuaciones Hiperbólicas

Las ecuaciones hiperbólicas también tratan con problemas de propagación como, por ejemplo, la ecuación de una onda, pero con la distinción que aparece una segunda derivada respecto al tiempo. En consecuencia, la solución consiste en distintos estados característicos con los cuales oscila el sistema, como es el caso de problemas de vibraciones, ondas de un fluido, transmisión de señales acústicas y eléctricas.

4.6. Métodos de Solución Numérica de las EDP

Las ecuaciones diferenciales en derivadas parciales, tanto las elípticas como las parabólicas e hiperbólicas, pueden ser resueltas sustituyendo y planteando distintos esquemas numéricos donde las derivadas parciales son reemplazadas por su aproximación en diferencias finitas divididas. A continuación, trataremos cada uno de los tres grupos antes mencionados.

4.6.1. Ecuaciones Elípticas

Para abordar la solución numérica de las ecuaciones elípticas utilizaremos como caso de estudio a la ecuación de Laplace, por ser utilizada en diversas áreas de las ciencias e ingeniería donde se trata con problemas que involucran la determinación de un potencial. Por ser un problema simple de plantear y resolver utilizaremos el caso de flujo de calor en régimen estacionario en una placa delgada. La expresión es:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \qquad (4.131)$$

Para abordar la solución numérica trataremos a la placa como una malla de puntos discretos (nodos) donde plantearemos la representación en diferencias finitas de la ecuación diferencial, lo cual transforma a la EDP en una ecuación algebraica en diferencias; utilizando diferencias finitas centradas de segundo orden podemos escribir:

$$\left[\frac{\partial^2 u}{\partial x^2}\right]_{ij} \approx \frac{1}{\Delta x^2}\left(u_{i-1j} - 2u_{ij} + u_{i+1j}\right) \tag{4.132}$$

$$\left[\frac{\partial^2 u}{\partial y^2}\right]_{ij} \approx \frac{1}{\Delta y^2}\left(u_{ij-1} - 2u_{ij} + u_{ij+1}\right)$$

Reemplazando estas expresiones en la EDP, queda:

$$\frac{1}{\Delta x^2}u_{i-1j} - 2\left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}\right)u_{ij} + \frac{1}{\Delta x^2}u_{i+1j} + \frac{1}{\Delta y^2}u_{ij-1} + \frac{1}{\Delta y^2}u_{ij+1} = 0 \tag{4.133}$$

Las condiciones de borde, o de frontera, deben estar especificadas para que exista una solución única. Existen dos posibilidades en cuanto a condiciones en la frontera:

- Especificar el valor de la función en el borde. Es la forma más simple y se la conoce como condición de frontera de Dirichlet o condición forzada.
- Especificar el valor de la derivada en la frontera. En general la derivada que se especifica es en la dirección normal al borde (flujo). Esta condición es conocida como condición de Neumann o condición natural.

4.6.2. Ecuaciones Parabólicas

Abordaremos el estudio de este tipo de ecuaciones mediante la solución de la ecuación de conducción del calor en una dimensión. No debe perderse de vista que los métodos que se desarrollarán a continuación son de aplicación a todas las ecuaciones que correspondan a esta clasificación. La ecuación de conducción de calor unidimensional es:

$$k\frac{\partial^2 T}{\partial x^2} = \frac{\partial T}{\partial t} \tag{4.134}$$

En esta ecuación, la función T es la temperatura que depende de x y de t, x es la variable independiente espacial, t es la variable independiente temporal y k es el coeficiente de difusividad térmica (cm^2/s).

En este caso hay que considerar que la solución presenta cambios en el espacio y en el tiempo. La malla usada para resolver por diferencias finitas la EDP, con dos variables independientes, puede ser representada por:

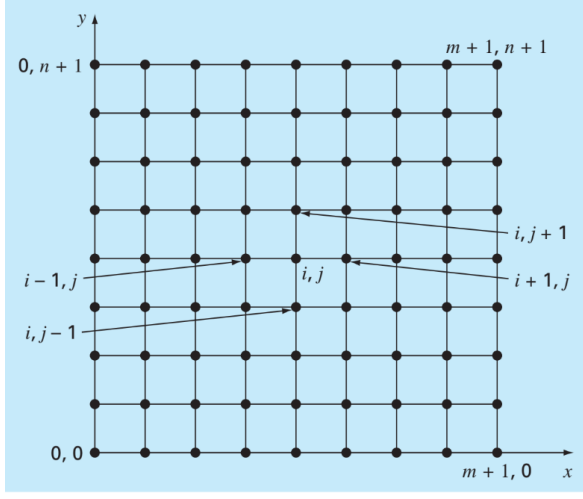


Figura 4.1.
Malla para solución EDP con dos variables (Chapra, 2017)

4.6.2.1. Método Explícito. La ecuación de conducción del calor que estamos tratando requiere dos aproximaciones. Para la segunda derivada, respecto de la variable espacial x, podemos hacerla con una diferencia dividida centrada con una aproximación de segundo orden:

$$\frac{\partial^2 T}{\partial x^2} \cong \frac{T_{i-1}^l - 2T_i^l + T_{i+1}^l}{(\Delta x^2)} \tag{4.135}$$

Y una diferencia dividida finita hacia delante para aproximar a la derivada en el tiempo:

$$\frac{\partial T}{\partial t} \cong \frac{T_i^{l+1} - T_i^l}{(\Delta t)} \tag{4.136}$$

De la aproximación adoptada para la variable x, utilizando operadores que corresponden a una interpolación limitada de segundo orden, surge que el error de truncamiento para x es del orden $O(\Delta x^3)$. De la misma forma, para la variable t utilizamos un operador que corresponde a una interpolación limitada de 1.er orden, en este caso surge el error de truncamiento para t de orden $O(\Delta t^2)$.

Sustituyendo en la ecuación:

$$k\frac{T_{i-1}^l - 2T_i^l + T_{i+1}^l}{(\Delta x^2)} = \frac{T_i^{l+1} - T_i^l}{(\Delta t)} \tag{4.137}$$

$$T_i^{l+1} = T_i^l + \left(\frac{k\Delta t}{(\Delta x)^2}\right)\left(T_{i-1}^l - 2T_i^l + T_{i+1}^l\right)$$

$$\lambda = \frac{k\,\Delta t}{(\Delta x)^2}$$

$$T_i^{l+1} = T_i^l + \lambda\left(T_{i-1}^l - 2T_i^l + T_{i+1}^l\right)$$

Esta ecuación, que puede ser escrita para todos los nodos interiores de la barra, proporciona un modo explícito para calcular los valores en cada nodo para un tiempo posterior con base en los valores actuales del nodo y sus vecinos. Esto puede ser esquematizado mediante la siguiente representación:

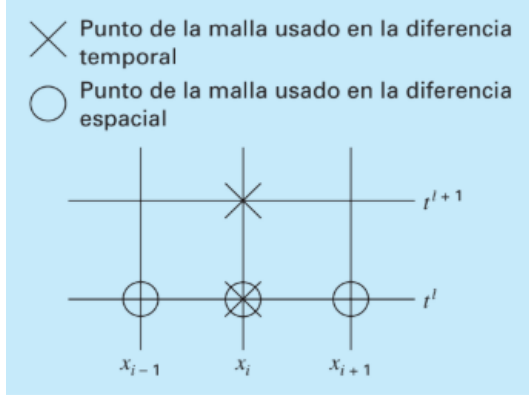


Figura 4.2.
Célula computacional de EDP del método explícito (Chapra, 2017)

4.6.2.2. Métodos Implícitos. Método implícito simple: Los métodos implícitos superan las dificultades de convergencia y estabilidad presentes en los métodos explícitos; proporcionan esquemas numéricos incondicionalmente estables, a expensas de usar algoritmos algo más complicados. El hecho de que sean incondicionalmente estables significa que la solución será estable para cualquier relación que exista entre Δx y Δt, a diferencia de los métodos explícitos que son condicionalmente estables. La diferencia fundamental entre ambas aproximaciones reside en que en la forma explícita aproximamos la derivada espacial en el nivel de tiempo l, de modo que nos quedaba una ecuación con una sola incógnita T_i^{l+1} , que podíamos despejar en forma explícita. En la forma implícita, la derivada espacial es aproximada en un nivel de tiempo posterior "l+1", de modo que queda más de una incógnita en una misma ecuación, impidiendo la resolución en forma sencilla como ocurría en el método explícito. Esta diferencia fundamental puede apreciarse claramente en la siguiente figura.

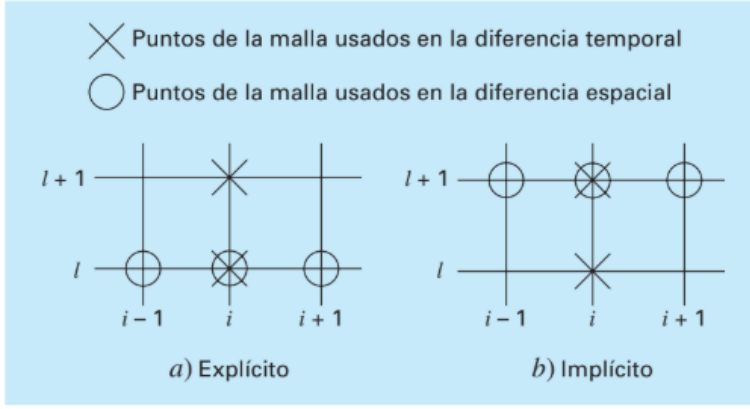


Figura 4.3.
Malla explícita e implícita de la solución de EDP (Chapra,2017)

Entonces, por quedar una ecuación con varias incógnitas, que no puede ser resuelta explícitamente, el sistema completo de ecuaciones que se originará debe resolverse simultáneamente. Esto es posible porque junto con las condiciones de frontera, las formas implícitas dan como resultado un conjunto de ecuaciones lineales algebraicas con el mismo número de incógnitas; entonces, el método se reduce a la resolución de un conjunto de ecuaciones simultáneas para cada instante de tiempo.

La ecuación de conducción de calor que estamos tratando requiere de dos aproximaciones. Para la segunda derivada, respecto a la variable espacial x, podemos hacerla con una diferencia dividida centrada con una aproximación de segundo orden, con el error de truncamiento que hemos discutido con anterioridad:

$$\frac{\partial^2 T}{\partial x^2} = \frac{T_{i+1}^{l+1} - 2 \cdot T_i^{l+1} + T_{i-1}^{l+1}}{(\Delta x)^2} \tag{4.138}$$

Y una diferencia dividida finita hacia delante para aproximar a la derivada en el tiempo:

$$\frac{\partial T}{\partial t} = \frac{T_i^{l+1} - T_i^l}{\Delta t} \tag{4.139}$$

Sustituyendo en la ecuación:

$$k \cdot \frac{\partial^2 T}{\partial x^2} = \frac{\partial T}{\partial t} \tag{4.140}$$

$$k\cdot\frac{T_{i+1}^{l+1}-2\cdot T_i^{l+1}+T_{i-1}^{+1l}}{(\Delta x)^2}=\frac{T_i^{l+1}-T_i^{l}}{\Delta t}$$

$$\lambda=\frac{k\cdot\Delta t}{(\Delta x)^2}$$

$$-\lambda\cdot T_{i-1}^{l+1}+(1+2\cdot\lambda)\cdot T_i^{l+1}-\lambda\cdot T_{i+1}^{l+1}=T_i^{l}$$

Esta ecuación se aplica en todos los nodos, excepto en el primero y el último. Para estos puntos valen las apreciaciones hechas en el caso anterior respecto de las condiciones de contorno. Vale destacar que el sistema de ecuaciones que se forma al aplicar este método es tridiagonal, y que existen algoritmos muy eficientes para la resolución de este tipo de sistemas como, por ejemplo, el método de Thomas.

Por otro lado, tenemos el método implícito de Crank-Nicolson que proporciona un esquema implícito de mayor exactitud que el método implícito simple visto anteriormente. Esto se logra desarrollando las aproximaciones por diferencias en el punto medio del incremento en el tiempo. Para hacer esto la primera derivada temporal puede ser aproximada en $t^{l+1/2}$ por:

$$\frac{\partial T}{\partial t}\cong\frac{1}{2}\cdot\left(\frac{T_i^{l+1}-T^{l+1/2}}{\Delta t/2}+\frac{T_i^{l+1/2}-T^{l}}{\Delta t/2}\right)=\frac{T_i^{l+1}-T_i^{l}}{\Delta t}\qquad(4.141)$$

La segunda derivada en el espacio puede ser determinada en el punto medio al promediar las aproximaciones por diferencias al inicio (t^l) y al final (t^{l+1}) del intervalo del incremento del tiempo:

$$\frac{\partial^2 T}{\partial x^2}\cong\frac{1}{2}\cdot\left[\frac{T_{i+1}^{l}-2\cdot T_i^{l}+T_{i-1}^{l}}{(\Delta x)^2}+\frac{T_{i+1}^{l+1}-2\cdot T_i^{l+1}+T_{i-1}^{l+1}}{(\Delta x)^2}\right]\qquad(4.142)$$

Sustituyendo en la ecuación de conducción de calor queda:

$$\frac{\Delta t\cdot k}{2\cdot(\Delta x)^2}\cdot\left[\left(T_{i+1}^{l}-2\cdot T_i^{l}+T_{i-1}^{l}\right)+\left(T_{i+1}^{l+1}-2\cdot T_i^{l+1}+T_{i-1}^{l+1}\right)\right]=T_i^{l+1}-T_i^{l}\qquad(4.143)$$

Esta ecuación se aplica en todos los nodos, excepto en el primero y el último. Para estos puntos valen las apreciaciones hechas en el caso anterior respecto de las condiciones de contorno. En las figuras siguientes puede apreciarse la diferencia entre las células computacionales del método implícito simple y el método implícito de Crank-Nicolson.

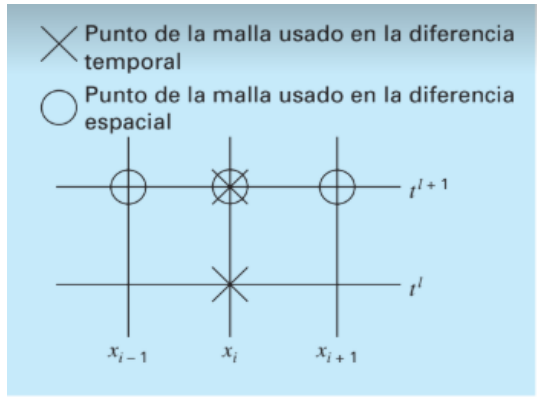


Figura 4.4.
Células computacionales del método implícito simple (Chapra, 2017)

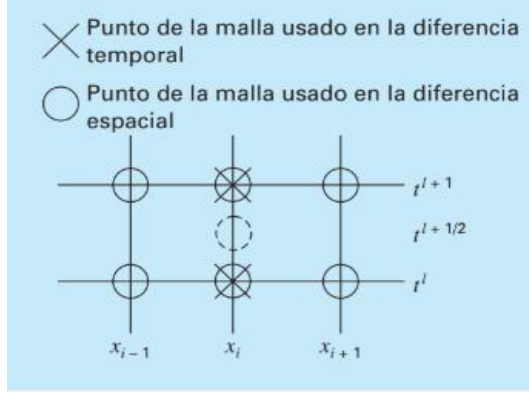


Figura 4.5.
Células computacionales del método implícito de Crank-Nicolson (Chapra, 2017)

4.6.3. Ecuaciones Hiperbólicas

Como hicimos en los casos anteriores, abordaremos el estudio del tratamiento de este tipo de ecuaciones mediante la resolución de una ecuación particular, pero no debe perderse de vista que los métodos que se desarrollarán a continuación son de aplicación a todas las ecuaciones que correspondan a esta clasificación. La ecuación a tratar en esta oportunidad es la ecuación de la onda unidimensional, cuya expresión es:

$$\frac{\partial^2 u}{\partial x^2} = a \frac{\partial^2 u}{\partial t^2} \tag{4.144}$$

Donde
$u = u(x,t)$ es la función de posición.

$\alpha = \frac{1}{c^2}$ Siendo c la velocidad de propagación en el medio.

En este caso, al igual que para las ecuaciones parabólicas, debemos conocer las condiciones iniciales del problema para poder hallar la solución. Entonces tendremos que:

$$u(x,0) = f(x) \qquad (4.145)$$
$$\left.\frac{\partial u}{\partial t}\right|_0 = g(x)$$

Como en el caso de las ecuaciones parabólicas pueden plantearse esquemas numéricos explícitos e implícitos.

4.6.3.1. Método Explícito. El esquema numérico que se obtiene en este caso surge de reemplazar las derivadas por su aproximación utilizando diferencias centrales en una interpolación limitada de segundo orden. Haciendo esto nos queda una expresión, en la cual se despeja apreciándose que la única incógnita es u_i^{l+1}.

$$\frac{1}{(\Delta x)^2}\left(u_{i-1}^{l} - 2\,u_i^{l} + u_{i+1}^{l}\right) = \frac{\alpha}{(\Delta t)^2}\left(u_i^{l-1} - 2\,u_i^{l} + u_i^{l+1}\right) \qquad (4.146)$$

$$u_i^{l+1} = \frac{(\Delta t)^2}{\alpha(\Delta x)^2}\left(u_{i-1}^{l} - 2\,u_i^{l} + u_{i+1}^{l}\right) + 2\,u_i^{l} - u_i^{l-1}$$

$$r^2 = \frac{(\Delta t)^2}{\alpha(\Delta x)^2}$$

$$u_i^{l+1} = r^2 \cdot \left(u_{i-1}^{l} - 2 \cdot u_i^{l} + u_{i+1}^{l}\right) + 2 \cdot u_i^{l} - u_i^{l-1}$$

$$u_i^{l+1} = r^2 \cdot u_{i-1}^{l} + \left(2 - 2 \cdot r^2\right) \cdot u_i^{l} + r^2 \cdot u_{i+1}^{l} + -u_i^{l-1}$$

Esta ecuación, que puede ser escrita para todos los nodos interiores del dominio, proporciona un modo explícito para calcular los valores en cada nodo para un tiempo posterior (nodo i en el tiempo l+1) con base en los valores actuales del nodo y sus vecinos (nodos i-1, 1 e i+1 en el tiempo l) y a un valor anterior del nodo considerado (nodo i en el tiempo l-1). Respecto de las condiciones de contorno, si estas son

del tipo forzada o Dirichlet, donde el valor de la función incógnita es conocido, la ecuación anterior no debe ser aplicada en los puntos de la frontera, puesto que allí no hay incógnitas. Si las condiciones de contorno son del tipo de Neumann (o condición natural) pueden ser incorporadas sin inconvenientes a las ecuaciones hiperbólicas mediante la utilización de una condición del tipo:

$$\frac{\partial u(0,t)}{\partial x} = m(0,t) \cong \frac{1}{2\cdot\Delta x}\cdot\left(-u_{i-1}^{l} + u_{i+1}^{l}\right) \qquad (4.147)$$

$$\frac{\partial u(l,t)}{\partial x} = n(l,t) \cong \frac{1}{2\cdot\Delta x}\cdot\left(-u_{i-1}^{l} + u_{i+1}^{l}\right)$$

Courant, Friedrichs y Lewy demostraron que este esquema numérico converge si se cumple con la condición: $0 < r \leq 1$. Donde r está definido por la expresión:

$$r^2 = \frac{(\Delta t)^2}{\alpha(\Delta x)^2} \qquad (4.148)$$

También se demostró que la estabilidad de la solución obtenida queda asegurada cuando se verifica que $r \leq 1$; entonces, la convergencia y estabilidad del esquema numérico explícito quedan aseguradas cuando se cumple con ambas condiciones en forma simultánea; es decir, cuando $0 < r \leq 1$.

4.6.3.2. Método Implícito. Para salvar los inconvenientes detallados anteriormente en el esquema explícito aplicado a este tipo de ecuaciones, es posible plantear un esquema implícito al costo de perder simplicidad en la solución, puesto que en este caso deberemos resolver un sistema de ecuaciones para hallar la solución. Siguiendo con el ejemplo de la ecuación de una onda, pero sin perder de vista que el procedimiento puede generalizarse para todas las ecuaciones de este tipo, tenemos que uno de los esquemas numéricos más utilizados es:

$$\frac{1}{(\Delta t)^2}\left(u_i^{l+1} - 2u_i^{l} + u_i^{l+1}\right) = \alpha\frac{1}{(\Delta x)^2}\left[\begin{array}{l}\frac{1}{4}\left(u_{ii-1}^{l+1} - 2u_{ii}^{l+1} + u_{ii+1}^{l+1}\right) + \\ \frac{1}{2}\left(u_{ii-1}^{l} - 2u_{ii}^{l} + u_{ii+1}^{l}\right) + \\ \frac{1}{4}\left(u_{ii-1}^{l-1} - 2u_{ii}^{l-1} + u_{ii+1}^{l-1}\right)\end{array}\right] \qquad (4.149)$$

Este se obtiene al reemplazar las derivadas por diferencias centrales en una interpolación limitada de segundo orden. Como vemos, en

este esquema la derivada segunda, respecto de x, se plantea como un promedio ponderado de la misma aplicada en los instantes de tiempo actual (l), anterior (l-1) y posterior (l+1). Este operador lleva a obtener un sistema de ecuaciones tridiagonales, y es incondicionalmente estable para todo valor de $r = \alpha \cdot \frac{\Delta t}{\Delta x}$.

4.7. Estudio de Casos

4.7.1. Ecuación de Laplace para una Placa Calentada

Formulación y solución del caso:

La ecuación general de Laplace en forma bidimensional puede ser escrita como:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \qquad (4.150)$$

Su aproximación en diferencias finitas será:

$$\frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{\Delta x^2} + \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{\Delta y^2} = 0 \qquad (4.151)$$

Para una malla cuadrada: $\Delta x = \Delta y$

$$T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1} - 4T_{i,j} = 0 \qquad (4.152)$$
$$T_{i,j} = (T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1})/4$$

En primera instancia trataremos el caso más simple, en el cual las temperaturas en las fronteras tienen un valor fijo, también podrían tratarse de cargas hidráulicas en un sistema acuífero. Esta es conocida como condición de frontera tipo Dirichlet. En caso ilustrado en la figura.

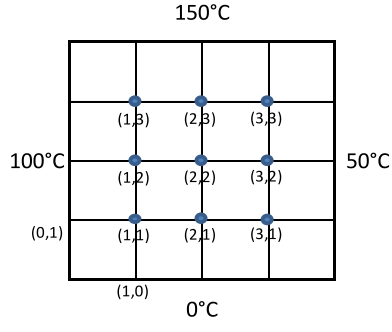


Figura 4.6.
Placa discretizada en 9 nudos y condiciones de borde o frontera

Para el caso presentado en la figura anterior; por ejemplo, para el nudo (1,1) se tiene la siguiente ecuación:

$$T_{2,1} + T_{0,1} + T_{1,2} + T_{1,0} - 4T_{1,1} = 0 \qquad (4.153)$$
$$T_{1,1} = (T_{2,1} + T_{0,1} + T_{1,2} + T_{1,0})/4$$

Sin embargo, T_{01}=75 °C y T_{10}=0 °C, por lo tanto, la ecuación anterior se puede expresar como:

$$-4T_{1,1} + T_{1,2} + T_{2,1} = -100 \qquad (4.154)$$
$$4T_{1,1} - T_{1,2} - T_{2,1} = 100$$

De la misma manera podemos establecer las ecuaciones para los 8 nudos restantes, y luego escribirlo como un sistema de ecuaciones de la siguiente manera:

4	-1		-1							$T_{1,1}$		100
-1	4	-1		-1						$T_{2,1}$		0
	-1	4	0		-1					$T_{3,1}$		50
-1		0	4	-1		-1				$T_{1,2}$		100
	-1		-1	4	-1		-1			$T_{2,2}$	=	0
		-1		-1	4	0		-1		$T_{3,3}$		50
			-1		0	4	-1			$T_{1,3}$		250
				-1		-1	4	-1		$T_{2,3}$		150
					-1		-1	4		$T_{3,3}$		200

Figura 4.7.
Sistema de ecuaciones en notación matricial

Solución numérica usando métodos matriciales: Para la solución numérica podemos emplear métodos matriciales ya utilizados anteriormente y resolver el sistema de ecuaciones. Completaremos la matriz de coeficientes con valores cero, calculamos la inversa y multiplicamos por el vector de coeficientes del lado derecho, obteniéndose:

4	-1	0	-1	0	0	0	0	0		$T_{1,1}$		100
-1	4	-1	0	-1	0	0	0	0		$T_{2,1}$		0
0	-1	4	0	0	-1	0	0	0		$T_{3,1}$		50
-1	0	0	4	-1	0	-1	0	0		$T_{1,2}$		100
0	-1	0	-1	4	-1	0	-1	0		$T_{2,2}$	=	0
0	0	-1	0	-1	4	0	0	-1		$T_{3,3}$		50
0	0	0	-1	0	0	4	-1	0		$T_{1,3}$		250
0	0	0	0	-1	0	-1	4	-1		$T_{2,3}$		150
0	0	0	0	0	-1	0	-1	4		$T_{3,3}$		200

Figura 4.8.
Sistema de ecuaciones en notación matricial completa

La inversa y multiplicación por el vector de coeficientes lado derecho da como resultado:

matriz inversa										matriz respuesta		
0.30	0.10	0.03	0.10	0.06	0.03	0.03	0.03	0.01		$T_{1,1}$		57.14
0.10	0.33	0.10	0.06	0.13	0.06	0.03	0.04	0.03		$T_{2,1}$		42.86
0.03	0.10	0.30	0.03	0.06	0.10	0.01	0.03	0.03		$T_{3,1}$		39.29
0.10	0.06	0.03	0.33	0.13	0.04	0.10	0.06	0.03		$T_{1,2}$		85.71
0.06	0.13	0.06	0.13	0.38	0.13	0.06	0.13	0.06		$T_{2,2}$	=	75.00
0.03	0.06	0.10	0.04	0.13	0.33	0.03	0.06	0.10		$T_{3,3}$		64.29
0.03	0.03	0.01	0.10	0.06	0.03	0.30	0.10	0.03		$T_{1,3}$		110.71
0.03	0.04	0.03	0.06	0.13	0.06	0.10	0.33	0.10		$T_{2,3}$		107.14
0.01	0.03	0.03	0.03	0.06	0.10	0.03	0.10	0.30		$T_{3,3}$		92.86

Figura 4.9.
Sistema de ecuaciones en notación matricial, matriz inversa y matriz respuesta

Solución numérica usando métodos iterativos (Método de Liebman): La mayoría de soluciones numéricas de la ecuación de Laplace involucran sistemas que son mucho más grandes; así el mismo ejemplo se puede resolver usando una malla mayor como de 15 x 15, lo que resultaría en un sistema de 225 ecuaciones algebraicas lineales o 20 x 20 que daría 400 ecuaciones algebraicas lineales. Esta situación hace que una gran cantidad de ceros deben ser empleados, y por tanto se requiere gran cantidad de memoria computacional, en tal sentido se debe recurrir a métodos iterativos para resolver estas ecuaciones elípticas, como es el caso del método de Gauss-Seidel aplicado a las EDP, también conocido como el método de Liebmann.

Para esta técnica, la ecuación se resuelve de manera iterativa de j=1 a n y de i=1 a m. Debido a que la matriz que genera es diagonalmente dominante, este procedimiento convergirá en una solución estable. En algunos casos debemos emplear sobrerelajación para acelerar la razón de convergencia, aplicando la siguiente fórmula después de cada iteración.

$$T_{1,1} = (T_{2,1} + T_{0,1} + T_{1,2} + T_{1,0})/4 \tag{4.155}$$

$$T_{i,j}^{nuevo} = \lambda T_{i,j}^{nuevo} + (1-\lambda) T_{i,j}^{anterior}$$

Donde $T_{i,j}^{nuevo}$ y $T_{i,j}^{anterior}$ son los valores de $T_{i,j}$ de la iteración presente y la previa, respectivamente, y λ es un factor de peso que está entre 1 y 2.

Usando el método convencional de Gauss-Seidel, las iteraciones continúan hasta conseguir que los valores absolutos de todos los errores relativos porcentuales caigan dentro de los criterios preespecificados de parada.

$$(\varepsilon_a)_{i,j} = \left| \frac{T_{i,j}^{nuevo} - T_{i,j}^{anterior}}{T_{i,j}^{nuevo}} \right| .100\% \tag{4.156}$$

Este procedimiento podemos desarrollarlo en forma manual utilizando la hoja de cálculo para iteraciones, paso a paso, o aplicando el proceso iterativo que brinda la misma hoja de cálculo. Para esto activamos opciones de Excel, fórmulas y habilitamos el cálculo iterativo, donde podemos especificar el número máximo de iteraciones y el valor del error o condición de parada en el proceso de iteración.

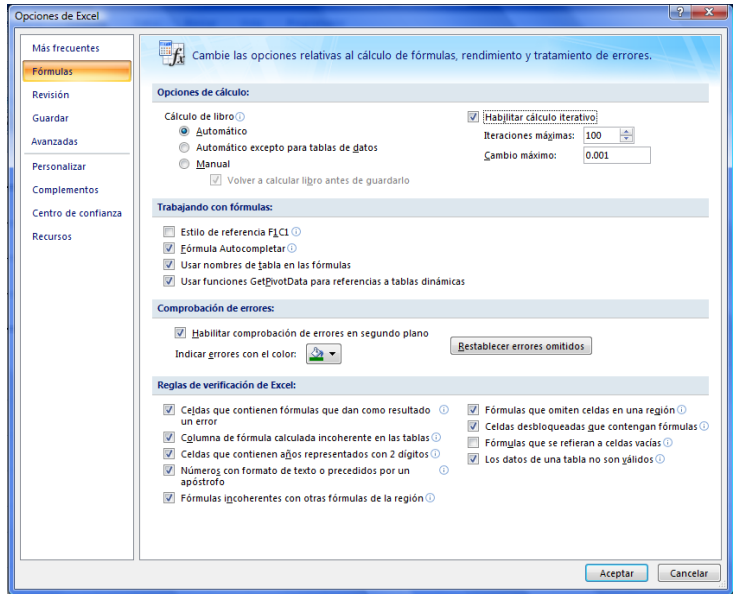


Figura 4.10.
Configuración Excel para el proceso iterativo

Activada esta opción, podemos formular la ecuación de Laplace sin que se anuncien errores cíclicos y obtener los siguientes resultados para el caso de la malla con 3 x 3.

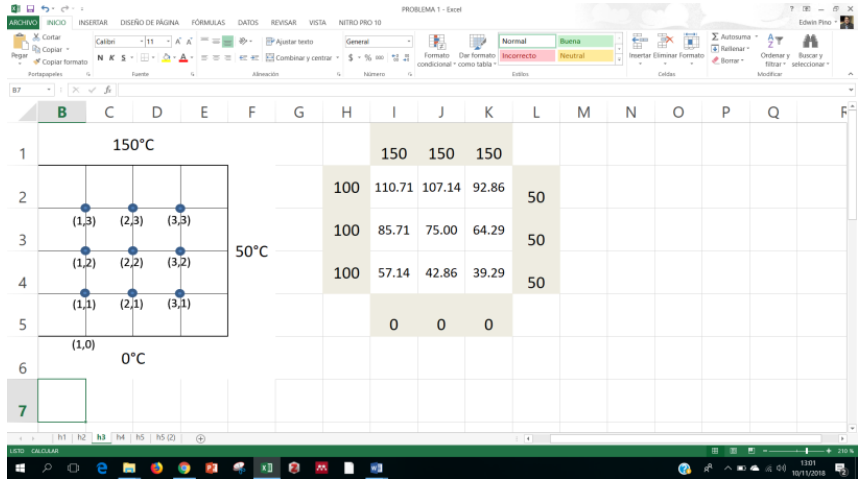


Figura 4.11.
Solución Excel de malla de 3 x 3

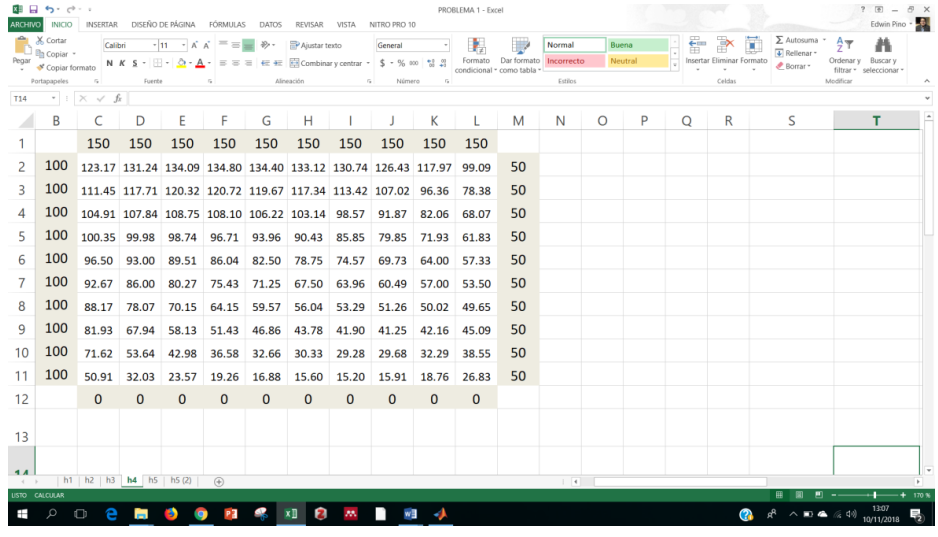
Si utilizamos el código MATLAB podemos establecer:

```
clc;
clear;
a=[4   -1  0  -1 0      0      0      0      0;
  -1  4  -1 0  -1 0      0      0      0;
   0   -1  4     0      0  -1  0      0      0;
  -1  0     0     4  -1  0  -1  0      0;
   0   -1  0  -1 4  -1  0  -1  0;
   0  0  -1 0  -1 4      0      0  -1;
   0  0     0  -1 0      0      4  -1  0;
   0  0     0     0  -1  0  -1  4  -1;
   0  0     0     0      0  -1  0  -1  4];
b=[100 0 50 100 0 50 250 150 200]';
T=[1:9]';
c=a\b;
[T,c]
```

Lo cual arroja como resultados:

1.0000 57.1429
2.0000 42.8571
3.0000 39.2857
4.0000 85.7143
5.0000 75.0000
6.0000 64.2857
7.0000 110.7143
8.0000 107.1429
9.0000 92.8571

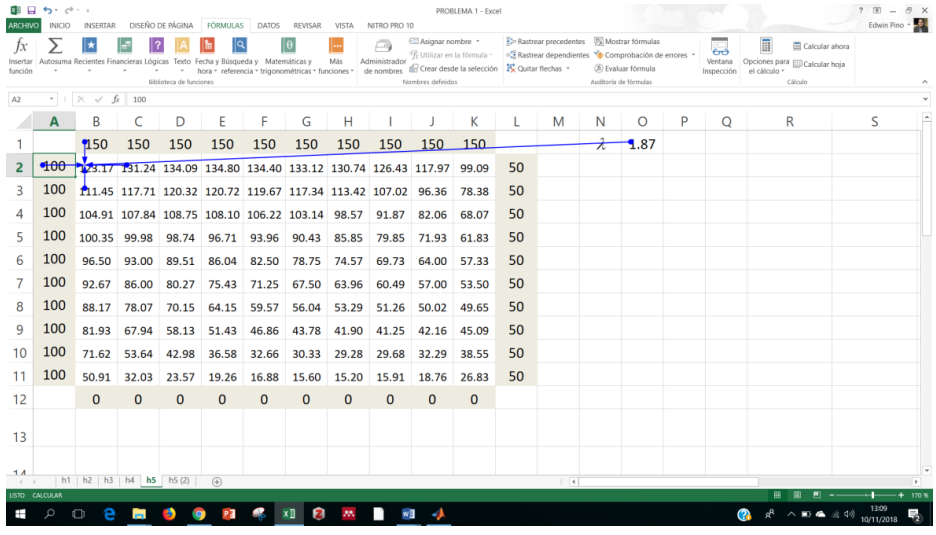
Al tratarse de un proceso bastante rápido y eficiente, podemos discretizar la malla a un número mayor de elementos, por ejemplo, 10 x 10; de lo cual obtenemos:



	B	C	D	E	F	G	H	I	J	K	L	M
1		150	150	150	150	150	150	150	150	150	150	
2	100	123.17	131.24	134.09	134.80	134.40	133.12	130.74	126.43	117.97	99.09	50
3	100	111.45	117.71	120.32	120.72	119.67	117.34	113.42	107.02	96.36	78.38	50
4	100	104.91	107.84	108.75	108.10	106.22	103.14	98.57	91.87	82.06	68.07	50
5	100	100.35	99.98	98.74	96.71	93.96	90.43	85.85	79.85	71.93	61.83	50
6	100	96.50	93.00	89.51	86.04	82.50	78.75	74.57	69.73	64.00	57.33	50
7	100	92.67	86.00	80.27	75.43	71.25	67.50	63.96	60.49	57.00	53.50	50
8	100	88.17	78.07	70.15	64.15	59.57	56.04	53.29	51.26	50.02	49.65	50
9	100	81.93	67.94	58.13	51.43	46.86	43.78	41.90	41.25	42.16	45.09	50
10	100	71.62	53.64	42.98	36.58	32.66	30.33	29.28	29.68	32.29	38.55	50
11	100	50.91	32.03	23.57	19.26	16.88	15.60	15.20	15.91	18.76	26.83	50
12		0	0	0	0	0	0	0	0	0	0	

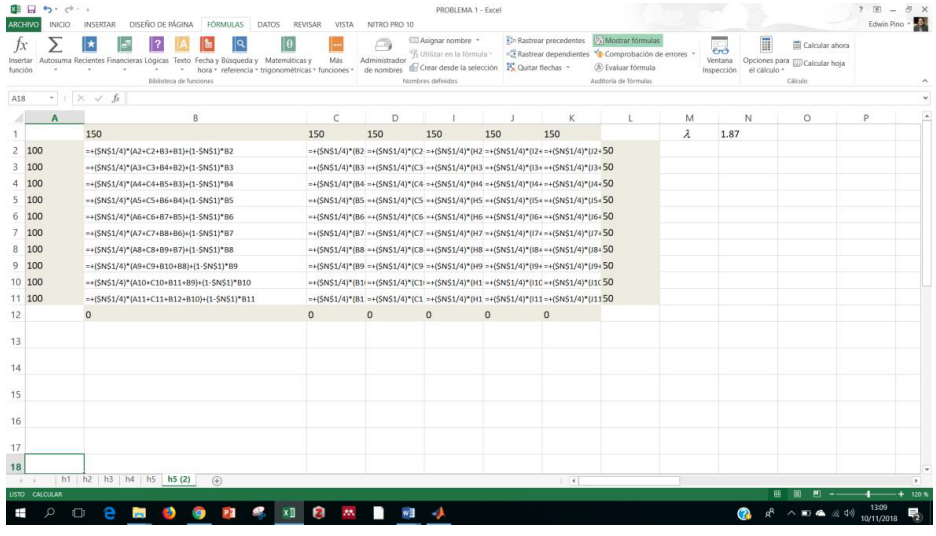
Figura 4.12.
Solución Excel de malla de 10 x 10

Ahora incluiremos el factor de sobrerelajación $\lambda = 1.87$ a la hoja de cálculo.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1		150	150	150	150	150	150	150	150	150	150			λ	1.87				
2	100	123.17	131.24	134.09	134.80	134.40	133.12	130.74	126.43	117.97	99.09	50							
3	100	111.45	117.71	120.32	120.72	119.67	117.34	113.42	107.02	96.36	78.38	50							
4	100	104.91	107.84	108.75	108.10	106.22	103.14	98.57	91.87	82.06	68.07	50							
5	100	100.35	99.98	98.74	96.71	93.96	90.43	85.85	79.85	71.93	61.83	50							
6	100	96.50	93.00	89.51	86.04	82.50	78.75	74.57	69.73	64.00	57.33	50							
7	100	92.67	86.00	80.27	75.43	71.25	67.50	63.96	60.49	57.00	53.50	50							
8	100	88.17	78.07	70.15	64.15	59.57	56.04	53.29	51.26	50.02	49.65	50							
9	100	81.93	67.94	58.13	51.43	46.86	43.78	41.90	41.25	42.16	45.09	50							
10	100	71.62	53.64	42.98	36.58	32.66	30.33	29.28	29.68	32.29	38.55	50							
11	100	50.91	32.03	23.57	19.26	16.88	15.60	15.20	15.91	18.76	26.83	50							
12		0	0	0	0	0	0	0	0	0	0								
13																			

Figura 4.13.
Solución incluyendo el factor de relajación



	A	B	C	D	I	J	K	L	M	N	O	P
1		150	150	150	150	150	150		λ	1.87		
2	100	=+($N$1/4)*(A2+C2+B3+B1)+(1-$N$1)*B2	=+($N$1/4)*(B2	=+($N$1/4)*(C2	=+($N$1/4)*(H2	=+($N$1/4)*(I2+	=+($N$1/4)*(J2+	50				
3	100	=+($N$1/4)*(A3+C3+B4+B2)+(1-$N$1)*B3	=+($N$1/4)*(B3	=+($N$1/4)*(C3	=+($N$1/4)*(H3	=+($N$1/4)*(I3+	=+($N$1/4)*(J3+	50				
4	100	=+($N$1/4)*(A4+C4+B5+B3)+(1-$N$1)*B4	=+($N$1/4)*(B4	=+($N$1/4)*(C4	=+($N$1/4)*(H4	=+($N$1/4)*(I4+	=+($N$1/4)*(J4+	50				
5	100	=+($N$1/4)*(A5+C5+B6+B4)+(1-$N$1)*B5	=+($N$1/4)*(B5	=+($N$1/4)*(C5	=+($N$1/4)*(H5	=+($N$1/4)*(I5+	=+($N$1/4)*(J5+	50				
6	100	=+($N$1/4)*(A6+C6+B7+B5)+(1-$N$1)*B6	=+($N$1/4)*(B6	=+($N$1/4)*(C6	=+($N$1/4)*(H6	=+($N$1/4)*(I6+	=+($N$1/4)*(J6+	50				
7	100	=+($N$1/4)*(A7+C7+B8+B6)+(1-$N$1)*B7	=+($N$1/4)*(B7	=+($N$1/4)*(C7	=+($N$1/4)*(H7	=+($N$1/4)*(I7+	=+($N$1/4)*(J7+	50				
8	100	=+($N$1/4)*(A8+C8+B9+B7)+(1-$N$1)*B8	=+($N$1/4)*(B8	=+($N$1/4)*(C8	=+($N$1/4)*(H8	=+($N$1/4)*(I8+	=+($N$1/4)*(J8+	50				
9	100	=+($N$1/4)*(A9+C9+B10+B8)+(1-$N$1)*B9	=+($N$1/4)*(B9	=+($N$1/4)*(C9	=+($N$1/4)*(H9	=+($N$1/4)*(I9+	=+($N$1/4)*(J9+	50				
10	100	=+($N$1/4)*(A10+C10+B11+B9)+(1-$N$1)*B10	=+($N$1/4)*(B1	=+($N$1/4)*(C1	=+($N$1/4)*(H1	=+($N$1/4)*(I1C	=+($N$1/4)*(J1C	50				
11	100	=+($N$1/4)*(A11+C11+B12+B10)+(1-$N$1)*B11	=+($N$1/4)*(B1	=+($N$1/4)*(C1	=+($N$1/4)*(H1	=+($N$1/4)*(I11	=+($N$1/4)*(J11	50				
12		0	0	0	0	0	0					
13												
14												
15												
16												
17												
18												

Figura 4.14.
Formulación en Excel para el proceso iterativo

Una vez procesadas las temperaturas, según la malla que se muestra arriba, se procede a crear un archivo de datos tipo ".txt", haciendo uso del "Notepad" u otra aplicación, al cual hemos denominado "placa.txt", el cual contiene los resultados del proceso iterativo usando la hoja electrónica de cálculo.

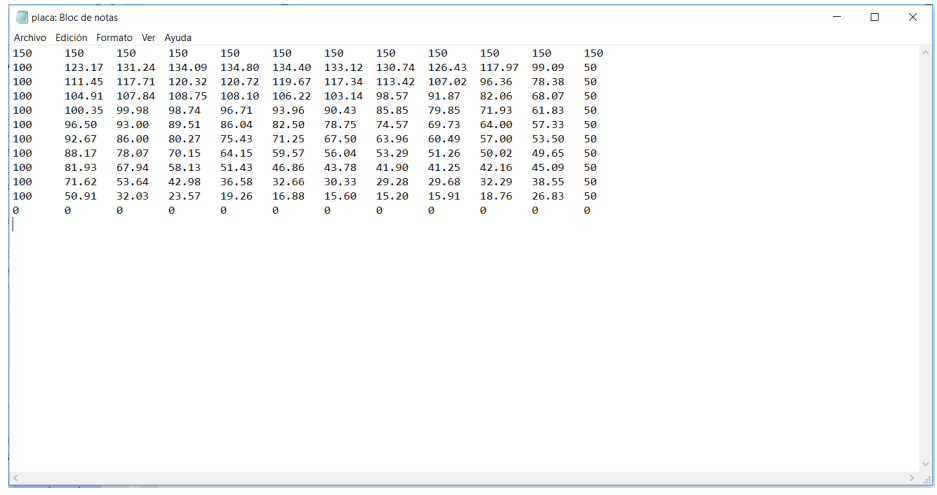


Figura 4.15.
Archivo .txt de datos finales

A continuación, elaboramos una codificación MATLAB para interpolar las isotermas y sus respectivas direcciones de flujo de calor:

```
clc;
clear;
A=load ('placa.txt'); % Se importa los datos de la matriz
respuesta
%Calculo del gradiente usando valores por defecto dx=1, dy=1,
por tanto seran correctas las direcciones y magnitudes
relativas
[px,py]=gradient(A);
%Contour, usamos contour para definir curvas de contorno o
isolineas
%Clabel, que agrega etiquetas de contorno a la grafica
%Quiver, toma los datos del gradiente y los agrega a la grafica
como flechas
curvas=contour(A);
clabel(curvas);
hold on
quiver(px,-py);
hold off
```

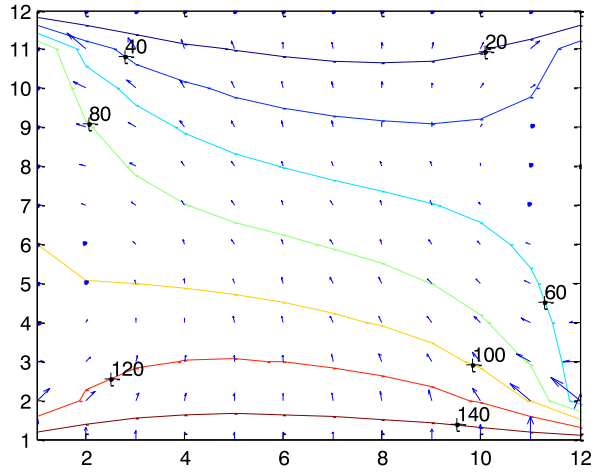


Figura 4.16.
Curvas de contorno obtenidas con el comando "contour" de MATLAB

Finalmente, proponemos el siguiente código computacional para la solución de la ecuación de Laplace para una placa calentada en el caso EDP elípticas.

```
%function ec_elip(a,b,h,tol,itermax)
a=10;b=10;h=1;tol=0.001;itermax=100;
n=a/h; m=b/h; k=0; emax=tol;
u=zeros(n,m);r=zeros(n,m);
u(:,1)=0; u(1,:)=75;
u(:,m)=100;u(n,:)=50;
while (k<itermax)&(emax>=tol)
   for j=2:1:(m-1)
   for i=2:1:(n-1)
      u(i,j)=u(i,j)+r(i,j);
      r(i,j)=(u(i+1,j)+u(i-1,j)+u(i,j+1)+u(i,j-1)-4*u(i,j))/4;
      if emax<r(i,j)
         emax=r(i,j);
      end
   end
end
k=k+1;
end
%surf(u)
plot(r)
r;
curvas=contour(u);
clabel(curvas);
```

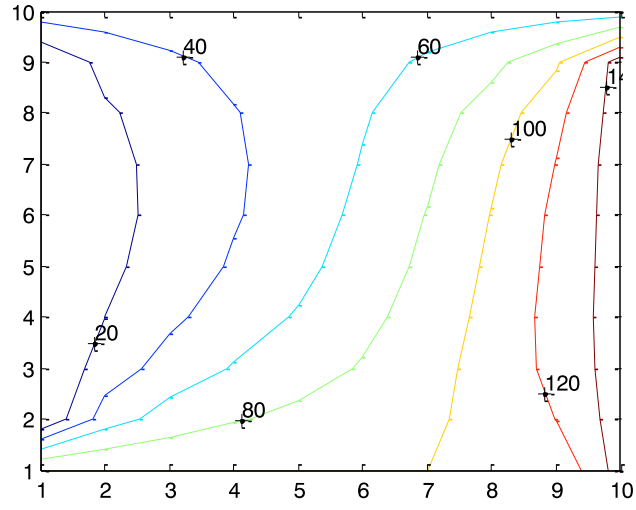


Figura 4.17.
Solución MATLAB para la placa calentada (2D)

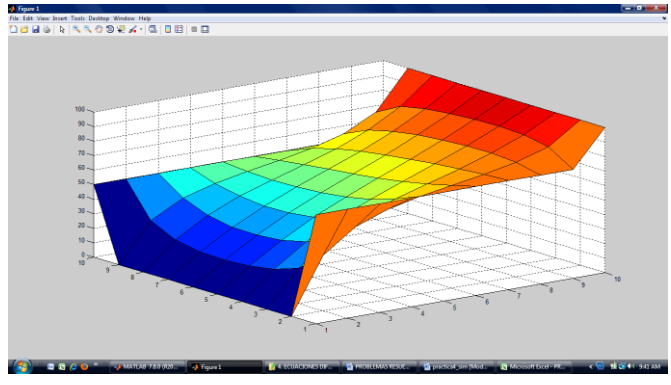


Figura 4.18.
Solución MATLAB para la placa calentada (3D)

4.7.2. Escurrimiento de un Fluido a través de un Medio Poroso

Formulación del caso:

La ecuación general que gobierna el escurrimiento de un fluido a través de un medio poroso bidimensional es:

$$K_x \frac{\partial^2 u}{\partial x^2} + K_y \frac{\partial^2 u}{\partial y^2} = 0 \qquad (4.157)$$

Donde:

K_X = Coeficiente de permeabilidad horizontal (cm/s)

K_Y = Coeficiente de permeabilidad vertical (cm/s)

u = Altura piezométrica = $p + y$

p/γ = Carga de presión del fluido circulante en cada punto (m.c.a.)
y = Carga de posición respecto de un plano de referencia cualquiera
γ = Densidad del fluido

(Se desprecia la carga por velocidad de la ecuación de Bernoulli por ser la velocidad de escurrimiento muy pequeña)

En diferencias finitas se tiene:

$$K_x \frac{1}{\Delta x^2}\left[u_{i,j-1} - 2u_{i,j} + u_{i,j+1}\right] + \ldots\ldots \qquad (4.158)$$
$$K_y \frac{1}{\Delta y^2}\left[u_{i-1,j} - 2u_{i,j} + u_{i+1,j}\right] = 0$$

Si colocamos puntos en el dominio, tal que formen una cuadrícula donde $\Delta x = \Delta y = \Delta$(malla cuadrada) y, además, suponemos que $K_X = K_Y = K$ (material isótropo), la expresión anterior se resume a lo siguiente:

$$u_{i-1\,i,j} - 4u_{i,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} = 0 \qquad (4.159)$$

Para el problema de la figura, además se requiere conocer las siguientes variables derivadas: (a) Distribución de presión del líquido en el suelo. (b) Distribución de velocidades de circulación del fluido (flujo). (c) Caudales de filtración en la sección media debajo de la estructura. (d) La distribución de presión en la base de la estructura.

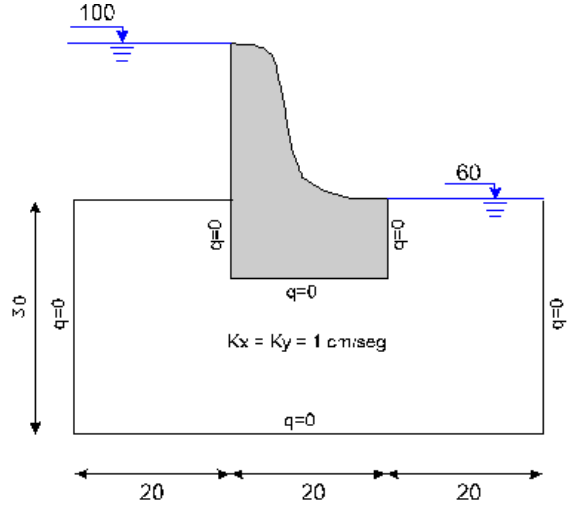


Figura 4.19.
Esquema de presa y volumen de control

Solución:

Adoptando $\Delta x = \Delta y = \Delta = 10$ m, subdividimos el dominio como se muestra en la siguiente figura:

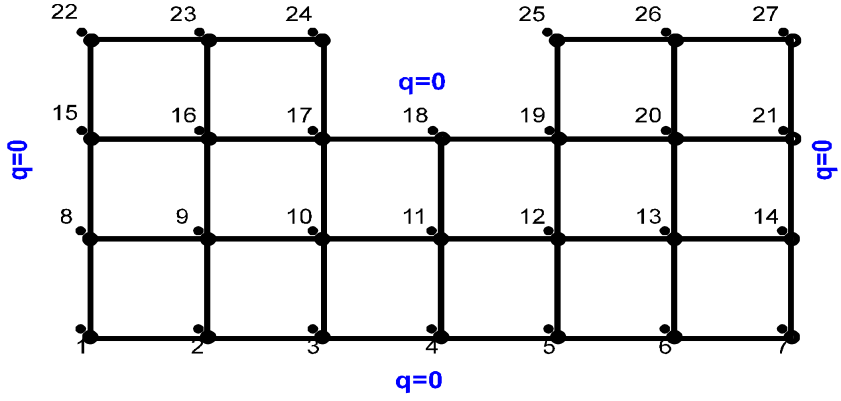


Figura 4.20.
Malla discretizada para el flujo debajo de una presa

Las condiciones de contorno forzadas (o Dirichlet) son:

$u22 = u23 = u24 = 100$
$u25 = u26 = u27 = 60$

La condición de pared impermeable (flujo nulo) indicada como qn=0 en las figuras anteriores corresponde a la condición de contorno natural (o Newman):

$$q_n = \frac{\partial u}{\partial n} = 0$$

Que para los puntos del contorno de la discretización son:

$qx_{17} = qx_{19} = (qx_{24} = qx_{25} =) 0$
$(qy_{17} =) qy_{18} = (qy_{19} =) 0$
$qy_1 = qy_2 = qy_3 = qy_4 = qy_5 = qy_6 = qy_7 = 0$
$qx_8 = qx_{15} = (qx_{22} =) qx_7 = qx_{14} = qx_{21} = (qx_{27} =) 0$

Obsérvese que en el punto de vértice tenemos más de una condición. En general la solución presenta alguna particularidad física en esos puntos.

Aplicado el operador diferencial en cada punto del dominio y contorno donde no se conoce el valor de *u*, se obtiene un sistema de ecuaciones algebraicas que en forma matricial es:

-4	2						2													u1		0
1	-4	1						2												u2		0
	1	-4	1						2											u3		0
		1	-4	1						2										u4		0
			1	-4	1						2									u5		0
				1	-4	1						2								u6		0
					2	-4							2							u7		0
1							-4	2						1						u8		0
	1						1	-4	1						1					u9		0
		1						1	-4	1						1				u10		0
			1						1	-4	1						1			u11	=	0
				1						1	-4	1						1		u12		0
					1						1	-4	1						1	u13		0
						1						2	-4							u14		0
							1							-4	2					u15		-100
								1						1	-4	1				u16		-100
									1						1	-4	1			u17		-100
										2						1	-4	1		u18		0
											1						1	-4	1	u19		-60
												1						1	-4	u20		-60
													1					2	-4	u21		-60

Figura 4.21.
Sistema de ecuaciones en notación matricial para la presa

En esta matriz no se han colocado los coeficientes nulos. Como se observa, la matriz de coeficientes resultante es una matriz banda, no simétrica. Resolviendo el sistema de ecuaciones obtenemos los valores de la variable incógnita en todos los puntos de la malla, siendo los valores obtenidos los indicados en la siguiente figura:

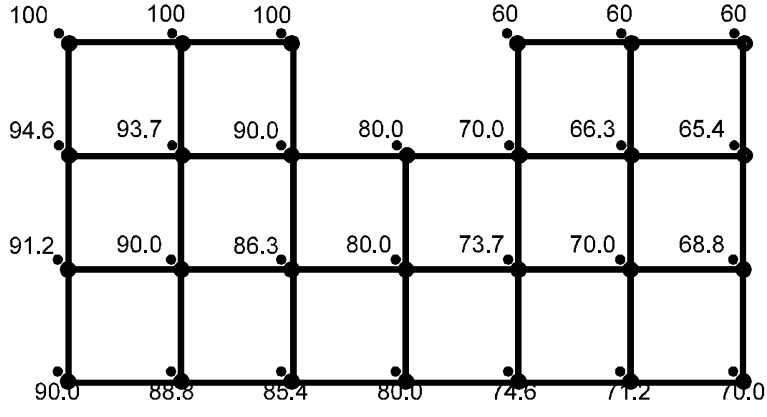


Figura 4.22.
Valores resultantes de las cargas hidráulicas en los nudos de la malla

De la misma manera podemos formular una hoja de cálculo en Excel activando antes la opción de cálculo iterativo para evitar errores de cálculo cíclico. Activamos la opción Archivo → Opciones → Fórmulas → y marcamos la opción "Habilitar cálculo iterativo" como se muestra en la siguiente figura.

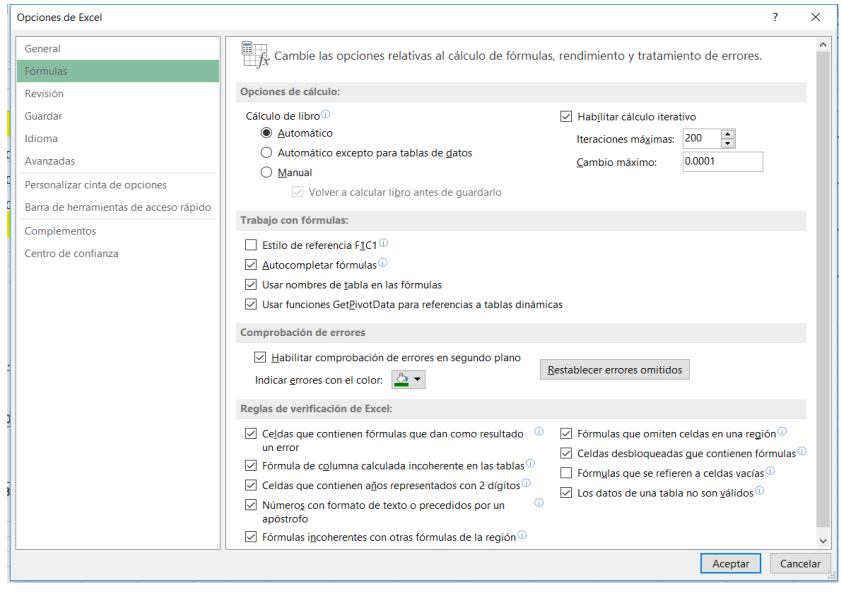


Figura 4.23.
Opción de Excel para el cálculo iterativo

A continuación, mostramos la formulación en Excel disponiendo celdas para las fórmulas respectivas, como ejemplo tenemos:

$$B2=B3:=(B1+B3+2*C2)/4$$
$$B4:=(2*B3+2*C4)/4$$
$$C3=D3=\ldots:=(C2+C4+B3+D3)/4$$

Las celdas en color azul representan las condiciones de borde tipo Dirichlet y las celdas amarillas, tipo Newman.

	A	B	C	D	E	F	G	H	I
1		100	100	100		60	60	60	
2		94.62	93.65	90.00	80.00	70.00	66.35	65.38	
3		91.15	90.00	86.35	80.00	73.65	70.00	68.85	
4		90.00	88.85	85.38	80.00	74.62	71.15	70.00	
5									

Figura 4.24.
Valores finales luego del proceso iterativo

Una forma gráfica habitual de presentar los resultados de un problema definido en un dominio bidimensional es mediante el trazado de las curvas de isopotencial o equipotenciales, donde cada curva corresponde a un valor de *u*=cte.

Una aproximación al trazado de las equipotenciales puede realizarse interpolando en la grilla los valores fijados para cada curva equipotencial. Así se han trazado las equipotenciales correspondientes a valores de *u* = 100, 95, 90, 85, 80, 75, 70, 65 y 60; en la siguiente figura:

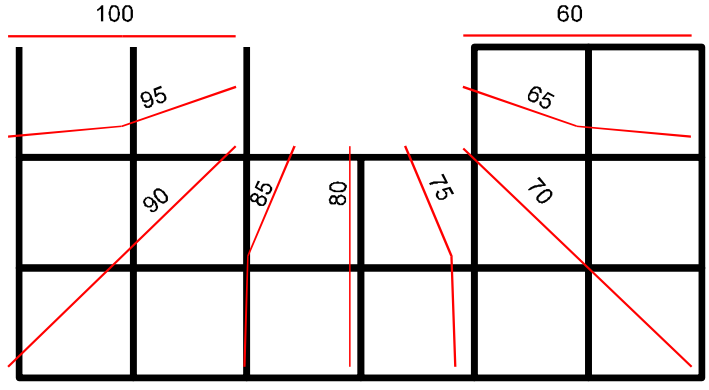


Figura 4.25.
Curvas de potenciales calculados

Aún resta encontrar los valores de las variables derivadas que son de interés en el problema. A esta etapa del proceso de solución numérica se le denomina post proceso de la solución.

Campo de velocidades: De acuerdo a la ley de Darcy, que gobierna el flujo en medios porosos, la velocidad de escurrimiento en cada punto resulta:

$$v_X = -K_X \frac{\partial u}{\partial x} \qquad ; \qquad v_Y = -K_Y \frac{\partial u}{\partial y}$$

Como ya conocemos los valores de *"u"* en cada uno de los puntos de la grilla es posible estimar, en esos mismos puntos, los componentes v_x y v_y, aplicando el operador central de la primera derivada y realizando las operaciones (procedimiento habitual para obtener la derivada de una función expresada en forma tabular). Para este caso resulta:

$$v_X = -K_X\left(-u_{i-1,j} + u_{i+1,j}\right) \qquad ; \qquad v_Y = -K_Y\left(-u_{i,j-1} + u_{1,j+1}\right)$$

Punto	Vx	Vy
1	0.00	0.00
2	0.12	0.00
3	0.22	0.00
4	0.27	0.00
5	0.22	0.00
6	0.12	0.00
7	0.00	0.00
8	0.00	-0.12
9	0.12	-0.12
10	0.25	-0.12
11	0.32	0.00

Punto	Vx	Vy
12	0.25	0.12
13	0.12	0.12
14	0.00	0.12
15	0.00	-0.22
16	0.12	-0.25
17	0.34	-0.34
18	0.50	0.00
19	0.34	0.34
20	0.12	0.25
21	0.00	0.97

Tabla 4.2.
Componentes de velocidad en los nudos de la malla

A modo de ejemplo se detalla el cálculo de los componentes de velocidad en el punto 10 de la grilla:

$$v_X = -K_X \left.\frac{\partial u}{\partial x}\right|_{10} \approx -\frac{1}{2\Delta x}\left[-u_9 + u_{11}\right] = 0.25\,\frac{cm}{s}$$

$$v_Y = -K_Y \left.\frac{\partial u}{\partial y}\right|_{10} = \approx -\frac{1}{2\Delta y}\left[-u_3 + u_{17}\right] = -0.12\,\frac{cm}{s}$$

En la figura siguiente se representa la distribución del campo de velocidades obtenido.

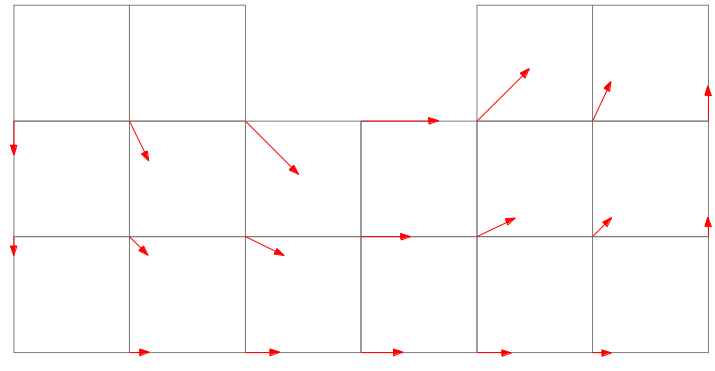


Figura 4.26.
Campo de velocidades

Diagrama de presión sobre la estructura del azud.
Sabemos que en este caso $p = u - y$, entonces para los puntos del azud en contacto con el medio poroso (suelo) resultan los siguientes valores de presión:

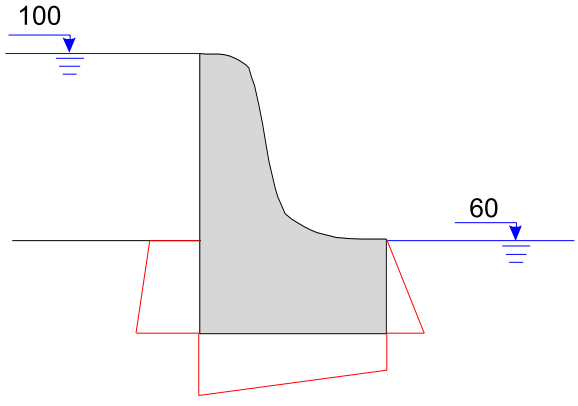


Figura 4.27.
Diagrama de distribución de presiones

Punto	Y	NP	P[kg/cm^2]
24	60	100	40
17	40	90.0	50
18	40	80.0	40
19	40	70.0	30
725	60	60	0

Tabla 4.3.
Presiones finales del diagrama de distribución de presiones

Cálculo del caudal de filtración: El caudal que atraviesa una sección S cualquiera se define como:

$$Q = \oint_S v_n \, . \, dS$$

Para conocer una estimación del caudal que escurre por debajo del azud se calculará el que atraviesa la sección vertical ubicada en el centro de la base. Esta sección resulta particularmente simple porque los vectores de velocidad solo tienen componentes en la dirección x y son perpendiculares a la sección elegida (simplificándose el cálculo).

Si se considera un área de ancho unitario en la dirección perpendicular a la figura (saliente del papel), la integral indicada (igual al área encerrada por el

perfil de velocidades en esta sección) puede obtenerse utilizando el método de los trapecios, resultando:

$$Q = \oint_S v.dS \approx \frac{(v_{18} + v_{11})}{2} x\,20 + \frac{(v_{11} + v_4)}{2} x\,20 \equiv 6.68 \frac{l}{s}$$

4.7.3. Ecuación de Conducción de Calor 1D

Formulación y solución del caso:

La ecuación que rige la conducción de calor unidimensional es:

$$k \cdot \frac{\partial^2 T}{\partial x^2} = \frac{\partial T}{\partial t} \tag{4.160}$$

$$\frac{\partial^2 T}{\partial x^2} \cong \frac{T_{i-1}^{l} - 2 \cdot T_i^{l} + T_{i+1}^{l}}{(\Delta x)^2}$$

$$\frac{\partial T}{\partial t} \cong \frac{T_i^{l+1} - T_i^{l}}{\Delta t}$$

$$k \cdot \frac{T_{i-1}^{l} - 2 \cdot T_i^{l} + T_{i+1}^{l}}{(\Delta x)^2} = \frac{T_i^{l+1} - T_i^{l}}{\Delta t}$$

De la aproximación adoptada para la variable *x*, utilizando operadores que corresponden a una interpolación limitada de segundo orden, surge que el error de truncamiento para *x* es del orden de $O(\Delta x^3)$. De la misma forma, para la variable *t*, donde utilizamos un operador que corresponde a una interpolación limitada de 1.er orden, surge que el error de truncamiento para *t* es del orden de $O(\Delta t^2)$.

Que puede ser expresada también como:

$$T_i^{l+1} = T_i^{l} + \left(\frac{k \cdot \Delta t}{(\Delta x)^2} \right) \cdot \left(T_{i-1}^{l} - 2 \cdot T_i^{l} + T_{i+1}^{l} \right) \tag{4.161}$$

Si hacemos $\lambda = \frac{k \cdot \Delta t}{(\Delta x)^2}$, nos queda:

$$T_i^{l+1} = T_i^{l} + \lambda \cdot \left(T_{i-1}^{l} - 2 \cdot T_i^{l} + T_{i+1}^{l} \right) \tag{4.162}$$

Esta ecuación, que puede ser escrita para todos los nodos interiores de la barra, proporciona un modo explícito para calcular los valores en cada nodo

para un tiempo posterior con base en los valores actuales del nodo y sus vecinos. Esto puede ser esquematizado mediante la siguiente representación de célula computacional para la forma explícita:

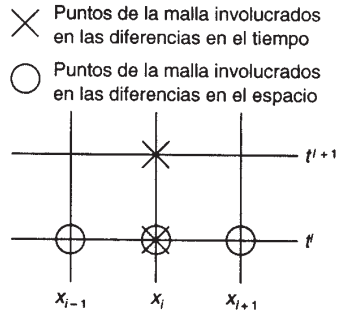


Figura 4.28.
Célula computacional en forma explícita (Chapra, 2005)

Si las condiciones de contorno son del tipo forzada o de Dirichlet, donde el valor de la función incógnita es conocido, la ecuación anterior no debe ser aplicada en los puntos de la frontera, puesto que allí no hay incógnitas. Las condiciones de contorno o de frontera del tipo de Neumann (o condición natural) pueden ser incorporadas sin inconvenientes a las ecuaciones parabólicas, de la misma manera que con las elípticas. En el caso particular de la ecuación de conducción de calor unidimensional deberán agregarse dos ecuaciones para caracterizar el balance de calor en los nodos extremos; por ejemplo, en el nodo inicial escribimos:

$$T_0^{l+1} = T_0^l + \lambda \cdot \left(T_{-1}^l - 2 \cdot T_0^l + T_1^l\right) \qquad q_x = -k \cdot \rho \cdot C \cdot \frac{\partial T}{\partial x} \tag{4.163}$$

$$\frac{\partial T}{\partial x} = \frac{T_{i-1}^l - T_{i+1}^l}{2 \cdot \Delta x}$$

$$q_x = -k \cdot \rho \cdot C \cdot \frac{\partial T}{\partial x} = -k \cdot \rho \cdot C \cdot \frac{T_{-1}^l - T_1^l}{2 \cdot \Delta x} \Rightarrow$$

$$T_{-1}^l = T_1^l - \frac{2 \cdot \Delta x \cdot q_x}{k \cdot \rho \cdot C}$$

$$T_0^{l+1} = T_0^l + \lambda \cdot \left(T_1^l - \frac{2 \cdot \Delta x \cdot q_x}{k \cdot \rho \cdot C} - 2 \cdot T_0^l + T_1^l \right) = ...$$

$$T_0^l + 2 \cdot \lambda \cdot \left(T_1^l - T_0^l - \frac{\Delta x \cdot q_x}{k \cdot \rho \cdot C} \right)$$

De la misma manera se puede obtener una ecuación para ser aplicada en el último punto. La solución explícita para la ecuación de conducción de calor unidimensional se basa en calcular la distribución de temperatura de una barra larga y delgada que tiene una longitud de 10 cm. El coeficiente de difusividad térmica es: $k = 0.835$ cm^2/s. Como condición de frontera tenemos que en los extremos de la barra la temperatura es constante todo el tiempo:

$T(0,t) = 100\,°C$ y $T(10,t) = 50\,°C$.

Como condición inicial tenemos que en el interior de la barra la temperatura para el tiempo $t = 0$ es:

$T(x,0) = 0\,°C$ para $0 < x < 10$.

Si tomamos $\Delta x = 1$ cm y $\Delta t = 0.1$ s tendremos que:

$$\lambda = \frac{k \cdot \Delta t}{(\Delta x)^2} = \frac{0.835 \cdot 0.1}{1^2} = 0.0835$$

Entonces aplicando la ecuación: $T_i^{l+1} = T_i^l + \lambda \cdot \left(T_{i-1}^l - 2 \cdot T_i^l + T_{i+1}^l\right)$

En la siguiente malla de diferencias:

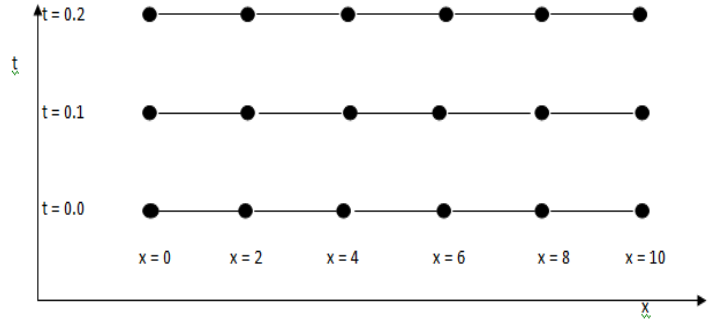


Figura 4.29.
Malla de diferencias finitas

Y así se continúa el cálculo, los resultados son mostrados con intervalos cada 3 segundos. Se observa que el aumento de temperatura con el tiempo representa la conducción de calor desde los extremos hacia la barra.

La solución en Excel se muestra a continuación:

$$T_i^{k+1} = T_i^k + \gamma\left(T_{i-1}^k - 2T_i^k + T_{i+1}^k\right)$$

dx	1
dt	0.1

a	0.835
g	0.084

T(i) 100
T(f) 50

$$\gamma = \frac{\alpha \Delta t}{\Delta x^2}$$

k	t	0	1	2	3	4	5	6	7	8	9	10
1	0.00	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	50.0
2	0.10	100.0	8.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.2	50.0
3	0.20	100.0	15.3	0.7	0.0	0.0	0.0	0.0	0.0	0.3	7.7	50.0
4	0.30	100.0	21.2	1.9	0.1	0.0	0.0	0.0	0.0	0.9	10.6	50.0
5	0.40	100.0	26.1	3.3	0.2	0.0	0.0	0.0	0.1	1.7	13.1	50.0
6	0.50	100.0	30.4	5.0	0.4	0.0	0.0	0.0	0.2	2.5	15.2	50.0
7	0.60	100.0	34.1	6.7	0.8	0.1	0.0	0.0	0.4	3.4	17.0	50.0
8	0.70	100.0	37.3	8.5	1.2	0.1	0.0	0.1	0.6	4.3	18.7	50.0
9	0.80	100.0	40.1	10.3	1.7	0.2	0.0	0.1	0.9	5.1	20.1	50.0
10	0.90	100.0	42.6	12.1	2.3	0.3	0.0	0.2	1.2	6.0	21.3	50.0
11	1.00	100.0	44.9	13.8	3.0	0.5	0.1	0.2	1.5	6.9	22.4	50.0
12	1.10	100.0	46.9	15.5	3.7	0.6	0.1	0.3	1.8	7.8	23.4	50.0
13	1.20	100.0	48.7	17.1	4.4	0.8	0.2	0.4	2.2	8.6	24.4	50.0
14	1.30	100.0	50.3	18.7	5.2	1.1	0.3	0.6	2.6	9.4	25.2	50.0
15	1.40	100.0	51.9	20.2	6.0	1.4	0.4	0.7	3.0	10.1	25.9	50.0
16	1.50	100.0	53.2	21.7	6.8	1.7	0.5	0.9	3.4	10.8	26.6	50.0

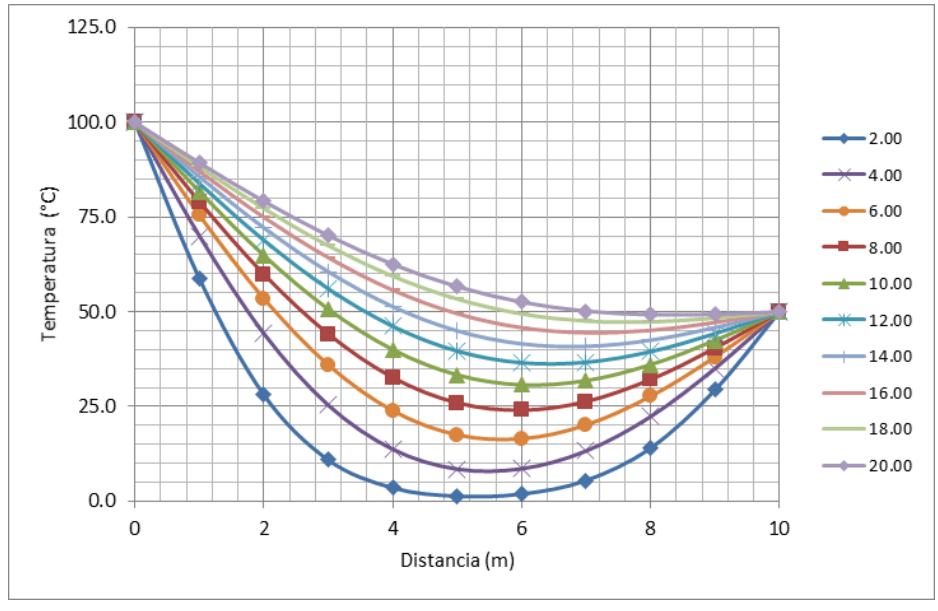


Figura 4.30.
Resultados en Excel de temperatura en función del tiempo y espacio en una barra delgada

El código computacional MATLAB se puede escribir de la siguiente manera:

```
clc; clear
D=10;
dx=1;
K=0.835;
```

```
dt=0.1;
L=K*dt/(dx^2);
B=[0:dx:D];
n=length(B);
To=100;
Tf=50;
Tm=0;
m=200;
U = Tm*ones(m,n);
for i=1:m
    U(i,1)=To;
    U(i,n)=Tf;
    for j=2:n-1
    U(i+1,j)=U(i,j)+L*(U(i,j+1)-2*U(i,j)+U(i,j-1));
    end
end

U(m+1,1)=To;
U(m+1,n)=Tf;

plot(B,U)
hold on
grid on
title('Ecuación de Calor')
```

El mismo que da como resultado:

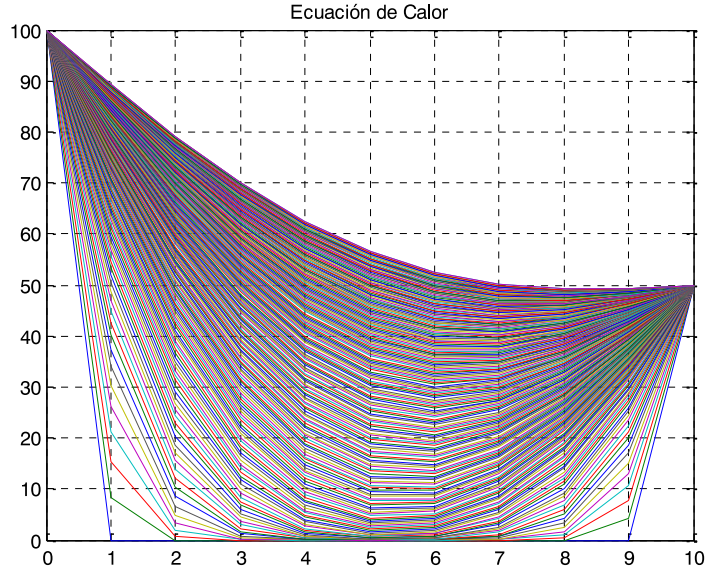


Figura 4.31.
Resultados en MATLAB de temperatura en función del tiempo y espacio en una barra delgada

4.7.4. Flujo en Acuífero Libre con Método Crank-Nicolson

Formulación y solución del caso:

Mediante el método Crank-Nicolson, de diferencias finitas, analizaremos la evolución temporal del nivel freático sobre una geometría de acuífero libre alterada por la excavación de una zanja. Las condiciones de contorno impuestas son de tipo Dirichlet (nivel piezométrico prescrito en el contorno). El objetivo principal es derivar y aplicar correctamente una formulación de Crank-Nicolson para resolverlo, haciendo hincapié en la estabilidad incondicional y la presencia de oscilaciones del método.

Se desea estudiar el efecto que produce una excavación sobre un terreno saturado homogéneo de permeabilidad K conocida. Para ello se plantea realizar un análisis de flujo en régimen no estacionario (transitorio), bajo una geometría de acuífero libre, tal y como se muestra en la siguiente figura.

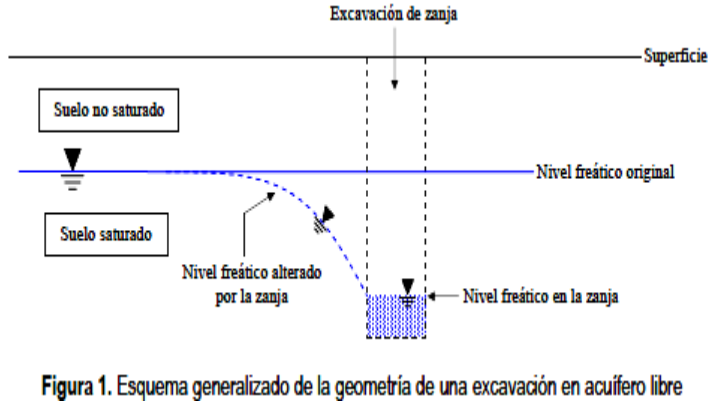


Figura 1. Esquema generalizado de la geometría de una excavación en acuífero libre

Figura 4.32.
Esquema generalizado de la geometría de una excavación en acuífero libre

La solución de un problema de flujo reside en encontrar el nivel piezométrico h y, en consecuencia, el caudal Q filtrado en la zanja debido al gradiente de presión de agua: $h = z + \frac{P_w}{\gamma}$, donde z representa la cota que tiene un punto del suelo, expresada en unidades de longitud sobre un nivel de referencia; P_w es la presión de agua expresada en unidades de fuerza por unidad de longitud; γ es el peso específico del agua (fuerza por unidad de superficie) y la relación P_w/γ representa la presión expresada en forma de columna de agua que tiene por encima dicho punto del suelo. Las siguientes figuras muestran un acuífero confinado y un acuífero libre.

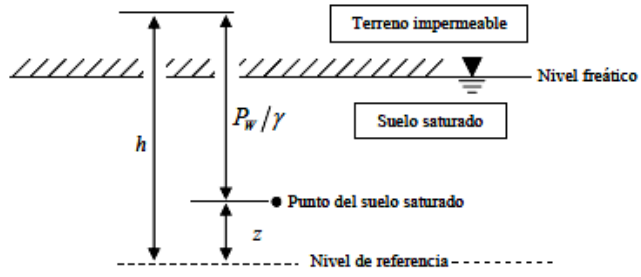


Figura 4.33.
Representación del nivel piezométrico con geometría de acuífero confinado

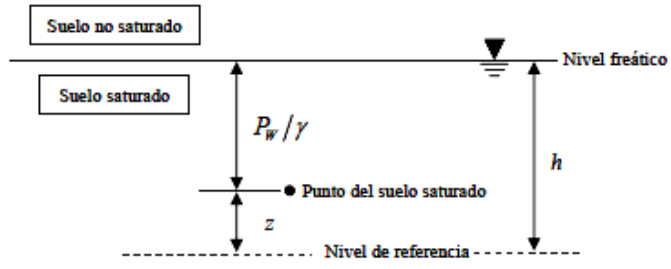


Figura 4.34.
Representación del nivel piezométrico con geometría de acuífero libre

Por tanto, el nivel piezométrico tiene unidades de longitud, y en este caso, debido a que no se tiene confinamiento, es igual al nivel freático (cota del nivel de agua que separa físicamente la parte saturada del terreno de la que no lo está), según muestra la figura anterior. El movimiento del agua con una geometría de acuífero libre (hipótesis de Dupuit), sin confinamiento, es por lo general bidimensional; es decir, esta se desplaza a través del suelo, tanto horizontal como verticalmente.

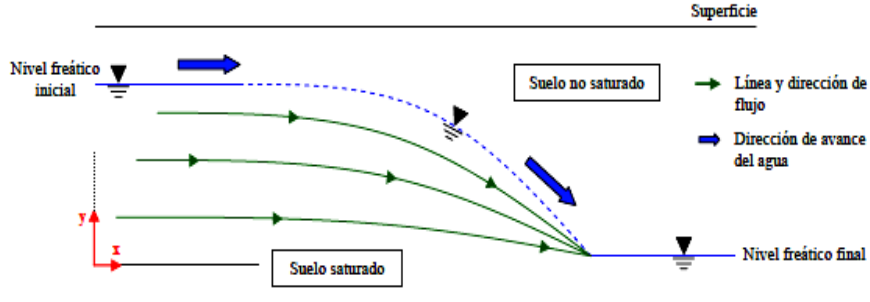


Figura 4.35.
Flujo de agua en una geometría de acuífero libre

Sin embargo, y bajo condiciones de líneas equipotenciales de nivel lo más verticales posible, la llamada hipótesis de Dupuit asume flujo solo horizontal y, por tanto, reduce el problema a uno estrictamente unidimensional. De este modo se puede considerar: $h(x, y) = h(x)$ y $Q(x, y) = Q(x)$.

Planteamiento físico: zanja en terreno saturado y heterogéneo.

Hipótesis básicas del problema:

- Se asume la hipótesis de Dupuit.
- Suelo homogéneo, isótropo y saturado.
- Análisis transitorio. Por tanto, tanto el nivel piezométrico como el caudal dependen también del tiempo: $h(x) = h(x,t)$ y $Q(x) = Q(x,t)$.

El movimiento del agua sobre el terreno viene caracterizado por la ley de flujo de Darcy, que muestra cómo esta fluye de mayor a menor nivel piezométrico:

$$q(x,t) = -K.\nabla h(x,t)$$

El problema general de flujo difusivo (sin considerar términos de reacción ni convección) en una geometría de acuífero libre está gobernado por la siguiente ecuación en derivadas parciales: $S_S\frac{\partial h(x,t)}{\partial t} = -\nabla.q(x,t) + r$, donde los parámetros del terreno que intervienen son:

- Recarga "r" (constante): Es el valor expresado en unidades de volumen por unidad de tiempo del caudal de agua que penetra al interior del acuífero producto de procesos activos en superficie. Depende, mayoritariamente, de la capacidad de infiltración del terreno.
- Coeficiente de almacenamiento específico "S_s" (constante): Hace referencia al volumen de agua liberado por una unidad de volumen de acuífero cuando el nivel piezométrico desciende una unidad. Está expresado en unidades del inverso de la longitud, y los valores típicos que se adoptan en acuíferos libres de espesor unitario varían entre 0.3 m^{-1} y 0.01 m^{-1}.

Aplicando la ley de Darcy se obtiene:

$$S_S\frac{\partial h(x,t)}{\partial t} = \nabla.(K.\nabla h(x,t) + r = K.\nabla.(\nabla h(x,t)) + r$$

Dado que el dominio espacial de la variable de estado $h(x,t)$ es unidimensional, el desarrollo del término de la divergencia $\nabla.[\nabla h(x,t)]$ en la expresión anterior se simplifica resultando la siguiente ecuación en derivadas parciales (EDP) parabólica 1D, que gobierna el problema planteado:

$$S_S\frac{\partial h(x,t)}{\partial t} = K\frac{\partial^2 h(x,t)}{\partial x^2} + r \qquad (4.164)$$

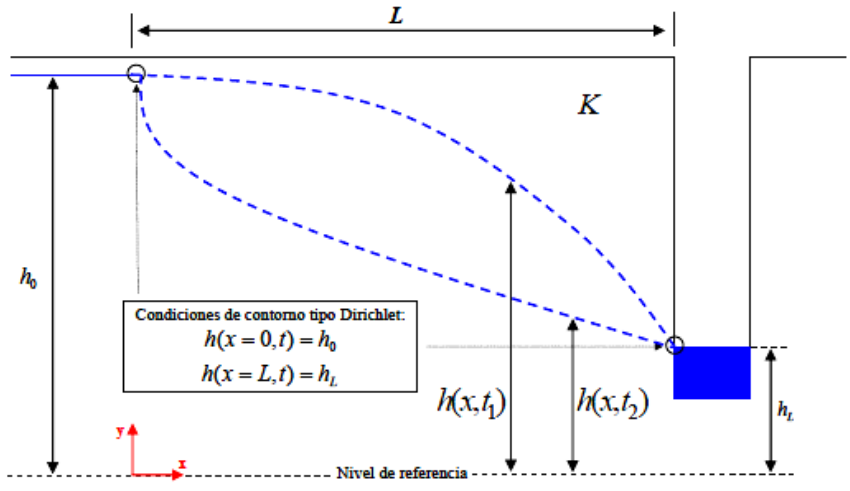


Figura 4.36.
Variables y parámetros de un esquema de excavación en terreno saturado y homogéneo

La solución requiere dos condiciones de contorno que, para este caso, se han propuesto tipo Dirichlet (valor de la variable preescrita): $h(x = 0,t) = h_0$ y $h(x = L,t) = h_L$; adicionalmente y el dominio de resolución, $x\epsilon[0, L]$; además, debido al carácter transitorio del problema, se necesita una condición inicial que haga referencia al estado del nivel piezométrico antes de ejecutar la excavación. En este caso, el nivel original es h_0 y, por tanto, la condición inicial es $h(x,t=0)=h_0$.

Problema Numérico: Esquema de Solución.

La siguiente EDP parabólica unidimensional con coeficiente (permeabilidad $K>0$) constante e invariable en el tiempo y condiciones de contorno de tipo Dirichlet se desea resolver mediante un esquema de diferencias finitas.

Término temporal **Término difusivo**

$$S_s\frac{\partial h(x,t)}{\partial t}=K\frac{\partial^2 h(x,t)}{\partial x^2}+r \quad \forall x\in[0,L] \quad \forall t\in[0,T]$$
$$h(x=0,t)=h_0$$
$$h(x=L,t)=h_L$$
$$h(x,t=0)=h_0 \qquad (4.165)$$

El objetivo es aproximar la ecuación en derivadas parciales a un problema discreto, cuya resolución conduce a un sistema lineal de ecuaciones en cada paso de tiempo. Para ello es necesario definir una discretización espacial (dirección x) y temporal (tiempo t).

$$x_i = x_0 + i\Delta x \qquad i = 0,1,\ldots,M+1$$
$$t^n = t_0 + n\Delta t \qquad n = 0,1,\ldots,N$$

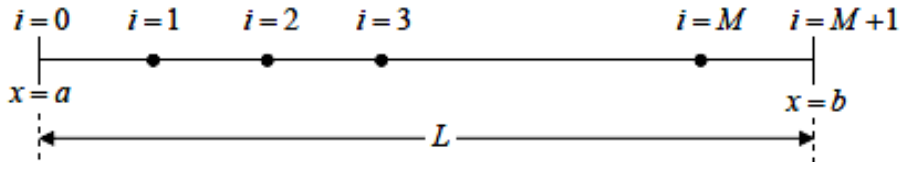


Figura 4.37.
Discretización del problema en la dirección x

A fin de resolver numéricamente la ecuación planteada en términos temporal y difusivo, se impone que esta se verifique en un punto $(x\,i, 1) = 1,\ldots,M$, y en un instante $t^{n+1/2}$, $n = 1,\ldots,$ N - 1, de la discretización. Para ello, de acuerdo con la siguiente figura, se denota el valor de la función h(x_i, $t^{n+1/2}$) como $hi^{n+1/2}$ aproximándola por $H_i^{n+1/2}$.

$$S_S \left.\frac{\partial h(x,t)}{\partial t}\right|_i^{n+1/2} = K \left.\frac{\partial^2 h(x,t)}{\partial x^2}\right|_i^{n+1/2} + r \qquad (4.166)$$

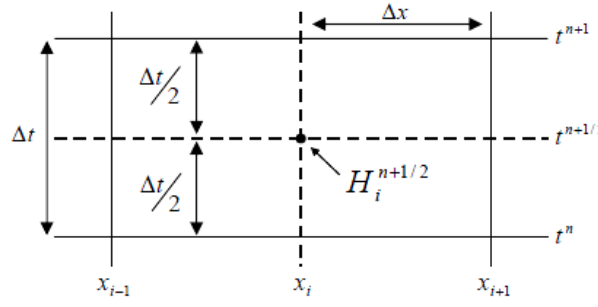


Figura 4.38.
Discretización del problema parabólico 1D

La derivada temporal de la ecuación anterior se aproxima mediante un esquema de diferencias finitas centrado con paso $\Delta t/2$. Asimismo, para aproximar el término difusivo (derivada espacial, no temporal) en un instante $t^{n+1/2}$ se interpola linealmente entre los instantes de tiempo t^n y t^{n+1}, de tal forma que se debe hallar ahora una discretización espacial para el término difusivo en los instantes de tiempo t^n y t^{n+1}. Para ello se aplica, tal como sigue, una aproximación centrada de segundo orden:

$$\left.\frac{\partial^2 h(x,t)}{\partial x^2}\right|_i^n = \frac{h_{i-1}^n - 2h_i^n + h_{i+1}^n}{\Delta x^2} + O(\Delta x^2) \tag{4.167}$$
$$\left.\frac{\partial^2 h(x,t)}{\partial x^2}\right|_i^{n+1} = \frac{h_{i-1}^{n+1} - 2h_i^{n+1} + h_{i+1}^{n+1}}{\Delta x^2} + O(\Delta x^2)$$

Por tanto, ahora para obtener la EDP discretizada tan solo hay que sustituir las expresiones, teniendo en cuenta la interpolación lineal del término difusivo;. sin embargo, para facilitar la lectura de la expresión resultante, se define el siguiente operador N:

$$N(h_i^n) = h_{i-1}^n - 2h_i^n + h_{i+1}^n \tag{4.168}$$

Resultando la siguiente ecuación para el problema discreto en diferencias finitas:

$$h_i^{n+1} - h_i^n = \frac{1}{2}RN\left(h_i^{n+1}\right) + \frac{1}{2}RN(h_i^n) + \frac{\Delta t}{S_s} rO(\Delta t^2, \Delta x^2) \tag{4.169}$$
$$i = 1, \ldots\ldots M, \ldots\ldots n > 0$$
$$h_o^n = h_o, \ldots\ldots n \geq 0$$
$$h_{M+1}^n = h_L, \ldots\ldots n \geq 0$$
$$h_i^o = h_o, \ldots\ldots i = 0, \ldots\ldots, M+1$$
$$R = \frac{K}{S_s}\frac{\Delta t}{\Delta x^2}$$

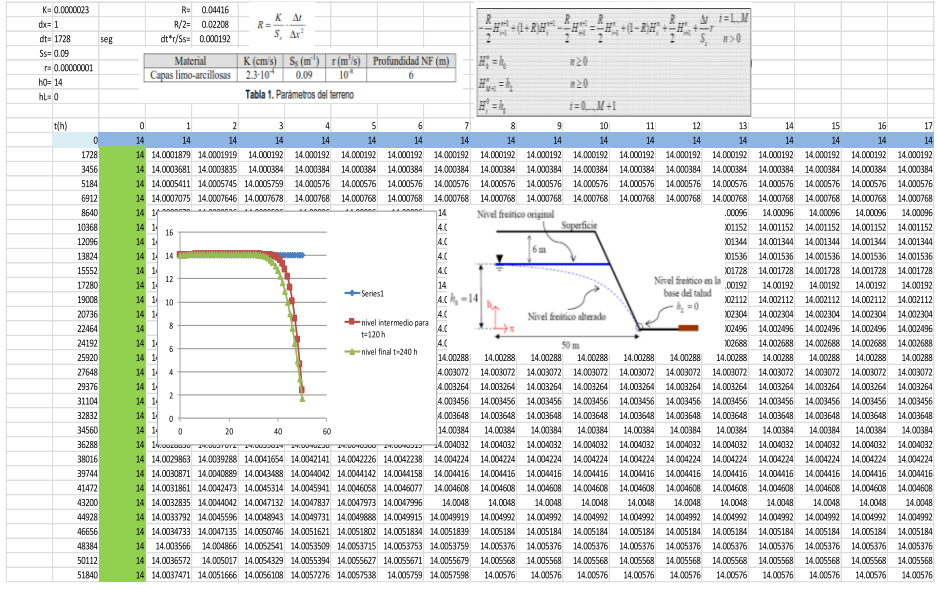
Nótese que la ecuación anterior cumple exactamente la EDP inicial ya que involucra el término del error. Eliminando dicho término se obtiene la aproximación discreta deseada:

$$h_i^{n+1} - \frac{1}{2}RN\left(h_i^{n+1}\right) = h_i^n + \frac{1}{2}RN(h_i^n) + \frac{\Delta t}{S_s} r \tag{4.170}$$
$$i = 1, \ldots\ldots M, \ldots\ldots n > 0$$
$$h_o^n = h_o, \ldots\ldots n \geq 0$$
$$h_{M+1}^n = h_L, \ldots\ldots n \geq 0$$
$$h_i^o = h_o, \ldots\ldots i = 0, \ldots\ldots, M+1$$

Substituyendo el operador N, la ecuación se puede reescribir como:

$$-\frac{R}{2}\left(h_{i-1}^{n+1}\right) + (1+R)h_i^{n+1} - \frac{R}{2}\left(h_{i+1}^{n+1}\right) = \frac{R}{2}\left(h_{i-1}^n\right) + (1-R)(h_i^n) + \frac{R}{2}\left(h_{i+1}^n\right) + \frac{\Delta t}{S_s} r \tag{4.171}$$
$$i = 1, \ldots\ldots M, \ldots\ldots n > 0$$
$$h_o^n = h_o, \ldots\ldots n \geq 0$$
$$h_{M+1}^n = h_L, \ldots\ldots n \geq 0$$
$$h_i^o = h_o, \ldots\ldots i = 0, \ldots\ldots, M+1$$

A continuación se muestra la solución en Excel para la ecuación formulada.



K= 0.0000023		R=	0.04416
dx= 1		R/2=	0.02208
dt= 1728	seg	dt*r/Ss=	0.000192
Ss= 0.09			
r= 0.00000001			
h0= 14			
hL= 0			

$$R = \frac{K}{S_s}\frac{\Delta t}{\Delta x^2}$$

Material	K (cm/s)	S_s (m^{-1})	r (m^2/s)	Profundidad NF (m)
Capas limo-arcillosas	$2.3\cdot10^{-4}$	0.09	10^{-8}	6

Tabla 1. Parámetros del terreno

$$-\frac{R}{2}H_{i-1}^{n+1}+(1+R)H_i^{n+1}-\frac{R}{2}H_{i+1}^{n+1}=\frac{R}{2}H_{i-1}^{n}+(1-R)H_i^{n}+\frac{R}{2}H_{i+1}^{n}+\frac{\Delta t}{S_s}\frac{r}{} \quad i=1\ldots M \quad n>0$$

$$H_1^n = h_0 \qquad n\geq 0$$

$$H_{M+1}^n = h_L \qquad n\geq 0$$

$$H_i^0 = h_i \qquad i=0,\ldots,M+1$$

Figura 4.39.
Solución Excel para ecuación parabólica 1D que gobierna el flujo en acuífero libre hacia un dren

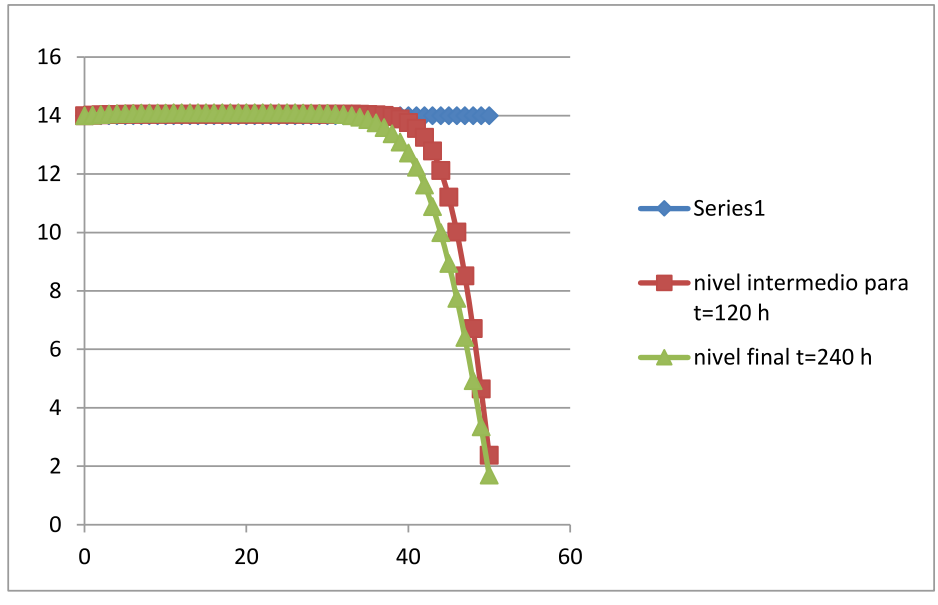


Figura 4.40.
Curvas de nivel freático a 120 h y 240 h

Asimismo, elaboramos un programa en MATLAB para la solución planteada.

```
clc;clear;
L=50;%input('ingrese longitd L:  ');
h1=14;%input('ingres nivel freatico inicial Ho:  ');
h2=0;%input('ingrese nivel freatico final Hf:  ');
t=240000;%input('ingrese tiempo de simulacion  T:');
dx=1;%input('variacion de longitud  dx:  ');
dt=100;%input('variacion de del tiempo  dt:  ');
k=0.00023;%input('ingrese coef.permeabilidad  k:  ');
Ss=0.09;%input('ingrese  coeficiente  de  alm.especifico   Ss:
');
r=0.00000001;%input('ingrese recarga hidraulica  r:  ');
R=(k*dt)/(Ss*dx^2);
C=(dt/Ss)*r;
B=[25:dx:L];
n=length(B);
H=h1*ones(t,n);
for i=1:t-1
    H(i,1)=h1;
    H(i,n)=h2;
    for j=2:n-1
        H(i+1,j)=((1-R)/(1+R))*H(i,j)+(R/(2*(1+R)))*(H(i,j-
1)+H(i,j+1)+H(i+1,j-1)+H(i+1,j+1))+C/(1+R);
    end
    H(i+1,n)=h2;
end

%disp(H);
plot(B,H)
%subplot(1,2,1); plot(B,H)
%subplot(1,2,2); plot (B,H)
%axis([9*L/10 L 4*h1/5 h1+1]);
title('simulacion de acuifero libre');
xlabel('longitud');
ylabel('nivel freatico');
```

Los resultados se muestran en la siguiente figura:

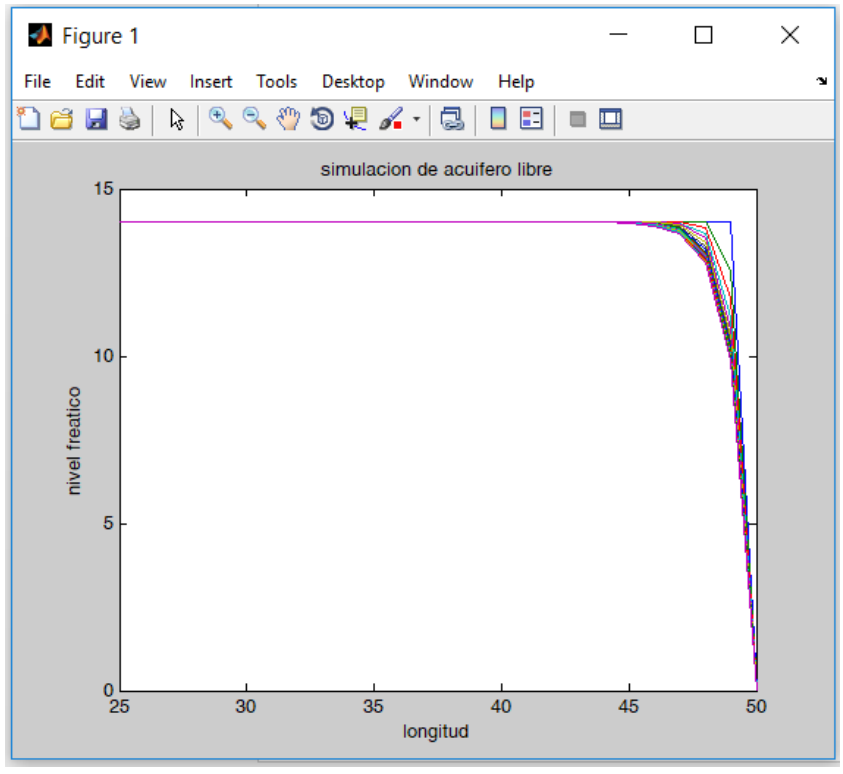


Figura 4.41.
Curvas de nivel freático obtenidas en MATLAB

4.7.5. Consolidación Unidimensional en Suelos con un Estrato Compresible

Formulación y solución del caso:

Para conocer la ruta de asentamiento de un suelo por cargas aplicadas, con base al ensayo de consolidación, se debe tomar en cuenta las siguientes hipótesis de Terzaghi (1925):

- Estrato de suelo homogéneo, isótropo y de espesor constante.
- Estrato saturado 100 % entre 1 o 2 superficies más permeables.
- Compresibilidad del agua y los granos, despreciable.
- Acciones similares de masas infinitesimales o masas grandes.
- Compresión unidimensional en dirección normal a la capa de suelo.
- Validez de la ley de Darcy.
- Valores constantes de las profundidades del suelo (algunas cambian).
- Relación lineal (idealizada) entre relación de vacíos y presión.
- Deformaciones lentas que permitan despreciar las fuerzas de inercia.

La modelación numérica en diferencias finitas usando el esquema numérico de Crank-Nicolson (1947) que es un esquema implícito alterno exacto en segundo orden, tanto en espacio como en tiempo, basado en la teoría de Terzaghi (1943), nos permite determinar la subsidencia de un suelo. Bajo esta premisa se realizó la simulación de la consolidación unidimensional con flujo de agua vertical para un estrato compresible, usando datos de campo y laboratorio. La ecuación que gobierna el proceso de consolidación unidimensional queda establecida de la siguiente manera:

$$\frac{\partial \mu}{\partial t} = Cv \frac{\partial^2 \mu}{\partial z^2} \tag{4.172}$$

Condición inicial:

$\mu = f(z)$, ……. para $t = 0$ y $0 < z < H$

Esta condición es debido al tipo de drenaje que se tenga; se atribuye a que las capas confinantes son suelos relativamente gruesos y el agua debe fluir libremente sin producirse sobre presiones intersticiales considerables.

Condición de frontera:
El estrato únicamente drena por la frontera superior.

$\boldsymbol{\mu = 0 \; en \; z = 0}$, para t > 0, $\boldsymbol{\mu = 0 \; en \; z = H}$

Esta última restricción se debe a que la base inferior es impermeable y, por lo tanto, no existe flujo a través de esta frontera, concluyéndose que el gradiente debe ser nulo.

La expresión discretizada inicial se establece de la siguiente manera:

$$\lambda/2\,\mu\_(i-1,j+1) - (\lambda+1)\,\mu\_(i-1,j+1) + \lambda/2\,\mu\_(i-1,j+1) = \lambda/2\,\mu\_(i-1,j+1) + (\lambda-1)\,\mu\_(i-1,j+1) - \lambda/2\,\mu\_(i-1,j+1) \tag{4.173}$$

Las condiciones iniciales y de frontera se muestran en la siguiente figura:

$\mu\_(i,0) = f(z)$ para $i = 0, 1, \ldots$
$\mu\_(0,j) = 0$ para $j > 0$
$\mu\_(M,j) = 0$

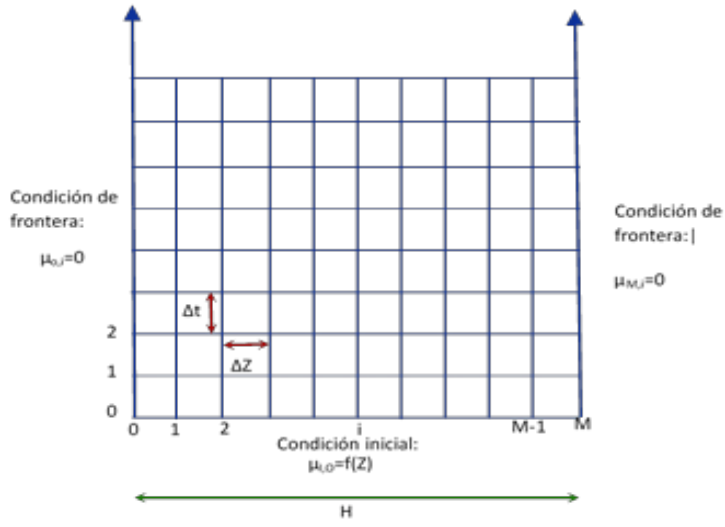


Figura 4.42.
Discretización del proceso unidimensional

Solución Numérica:

Si en la ecuación inicial hacemos $j = 0$ y realizamos un recorrido de los puntos $i = 1$ a $i =$ M-1, tomando las ecuaciones anteriores, podemos escribir las ecuaciones en la línea del tiempo.

$$(1+\lambda)\mu_{1,1} - \frac{\lambda}{2}\mu_{2,1} = (1-\lambda)\mu_{1,0} + \frac{\lambda}{2}(\mu_{0,1} + \mu_{0,0} + \mu_{2,0}) \tag{4.174}$$

$$-\frac{\lambda}{2}\mu_{1,1} + (1+\lambda)\mu_{2,1} - \frac{\lambda}{2}\mu_{3,1} = (1-\lambda)\mu_{2,0} + \frac{\lambda}{2}(\mu_{1,0} + \mu_{3,0})$$

$$-\frac{\lambda}{2}\mu_{2,1} + (1+\lambda)\mu_{3,1} - \frac{\lambda}{2}\mu_{4,1} = (1-\lambda)\mu_{3,0} + \frac{\lambda}{2}(\mu_{2,0} + \mu_{4,0})$$

.....

$$-\frac{\lambda}{2}\mu_{i-1,1} + (1+\lambda)\mu_{i,j} - \frac{\lambda}{2}\mu_{i+1,1} = (1-\lambda)\mu_{i,0} + \frac{\lambda}{2}(\mu_{i-1,0} + \mu_{i+1,0})$$

.....

$$-\frac{\lambda}{2}\mu_{M-3,1} + (1+\lambda)\mu_{M-2,1} - \frac{\lambda}{2}\mu_{M-1,1} = (1-\lambda)\mu_{M-2,0} + \frac{\lambda}{2}(\mu_{M-3,0} + \mu_{M-1,0})$$

$$-\frac{\lambda}{2}\mu_{M-2,1} + (1+\lambda)\mu_{M-1,1} - \frac{\lambda}{2}\mu_{M-1,1} = (1-\lambda)\mu_{M-1,0} + \frac{\lambda}{2}(\mu_{M-2,0} + \mu_{M,0})$$

Se tiene ecuaciones lineales simultáneas con M-1 incógnitas. Las incógnitas son $\mu i,1$ para i=1 a i=M-1 y dejando a las incógnitas en el lado izquierdo de las igualdades. El sistema resultante es tridiagonal, porque los elementos son diferentes a cero. El método a usar será el método de eliminación de Gauss. El método de eliminación de Gauss se puede expresar de la siguiente forma:

$$
\begin{aligned}
&b_1\mu_{1,1} + c_1\mu_{2,1} = d_1 \\
&a_2\mu_{1,1} + b_2\mu_{2,1} + c_2\mu_{3,1} = d_2 \\
&a_3\mu_{2,1} + b_3\mu_{3,1} + c_3\mu_{4,1} = d_3 \\
&\ldots\ldots \\
&a_i\mu_{i-1,1} + b_i\mu_{i,1} + c_i\mu_{i+1,1} = d_i \\
&\ldots\ldots \\
&a_{M-2}\mu_{M-3,1} + b_{M-2}\mu_{M-2,1} + c_{M-2}\mu_{M+1} = d_{M-2} \\
&a_{M-1}\mu_{M-2,1} + b_{M-2}\mu_{M-1,1} = d_{M-1}
\end{aligned}
\tag{4.175}
$$

Donde:

(4.176)

$$
\begin{aligned}
&b_1 = 1+\lambda \quad c_1 = -\lambda/2 \quad d_1 = (1-\lambda)\mu_{1,0} + \lambda/2\ (\mu_{0,1} + \mu_{0,0} + \mu_{2,0}) \\
&a_1 = -\lambda/2 \quad b_1 = 1+\lambda \quad c_1 = -\lambda/2 \\
&d_1 = (1-\lambda)\mu_{i,0} + \lambda/2\ (\mu_{0,1} + \mu_{i-1,0} + \mu_{i+1,0})
\end{aligned}
$$

Para $2 \leq i \leq$ M-2

$$
\begin{aligned}
&a_{M-1} = -\lambda/2 \quad b_{M-1} = 1+\lambda \\
&d_{M-1} = (1-\lambda)\mu_{M-1,0} + \lambda/2\ (\mu_{M,1} + \mu_{M-2,0} + \mu_{M,0})
\end{aligned}
\tag{4.177}
$$

Estas ecuaciones se abrevian de la siguiente forma:

$$
\begin{aligned}
&b_1\mu_{1,1} + c_1\mu_{2,1} = d_1 \\
&a_1\mu_{i-1,1} + b_i\mu_{i,1} + c_i\mu_{i+1,1} = d_i \ para\ 2 \leq i \leq M-2 \\
&a_{M-1}\mu_{M-2,1} + b_{M-1,1} = d_{M-1}
\end{aligned}
\tag{4.178}
$$

Habiendo definido las ecuaciones que gobiernan el proceso de consolidación, debemos tener en cuenta: $\mu i,o = \Delta P$ para i=1,2, …, M-1. La solución Excel para este caso se muestra a continuación:

$$\gamma = \frac{\alpha \Delta t}{\Delta x^2}$$

dx	0.5
dt	0.5

Cv	1.452
λ	2.904

H(i) 0
H(f) 0

i	t	0	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5
1	0.00	0	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.0
2	0.50	0	0.3049	0.4738	0.5649	0.6094	0.6227	0.609	0.565	0.474	0.305	0.0
3	1.00	0	0.1885	0.3369	0.4382	0.4959	0.5145	0.496	0.438	0.337	0.189	0.0
4	1.50	0	0.1351	0.2519	0.3393	0.3927	0.4106	0.393	0.339	0.252	0.135	0.0
5	2.00	0	0.1021	0.1929	0.2634	0.3078	0.3229	0.308	0.263	0.193	0.102	0.0
6	2.50	0	0.0787	0.1493	0.2048	0.2402	0.2524	0.24	0.205	0.149	0.079	0.0
7	3.00	0	0.061	0.1159	0.1594	0.1872	0.1968	0.187	0.159	0.116	0.061	0.0
8	3.50	0	0.0474	0.0902	0.1241	0.1458	0.1533	0.146	0.124	0.09	0.047	0.0
9	4.00	0	0.0369	0.0702	0.0966	0.1136	0.1194	0.114	0.097	0.07	0.037	0.0
10	4.50	0	0.0287	0.0547	0.0752	0.0884	0.093	0.088	0.075	0.055	0.029	0.0
11	5.00	0	0.0224	0.0426	0.0586	0.0689	0.0724	0.069	0.059	0.043	0.022	0.0
12	5.50	0	0.0174	0.0331	0.0456	0.0536	0.0564	0.054	0.046	0.033	0.017	0.0
13	6.00	0	0.0136	0.0258	0.0355	0.0417	0.0439	0.042	0.036	0.026	0.014	0.0
14	6.50	0	0.0106	0.0201	0.0277	0.0325	0.0342	0.033	0.028	0.02	0.011	0.0
15	7.00	0	0.0082	0.0156	0.0215	0.0253	0.0266	0.025	0.022	0.016	0.008	0.0
16	7.50	0	0.0064	0.0122	0.0168	0.0197	0.0207	0.02	0.017	0.012	0.006	0.0
17	8.00	0	0.005	0.0095	0.0131	0.0153	0.0161	0.015	0.013	0.009	0.005	0.0
18	8.50	0	0.0039	0.0074	0.0102	0.012	0.0126	0.012	0.01	0.007	0.004	0.0
19	9.00	0	0.003	0.0058	0.0079	0.0093	0.0098	0.009	0.008	0.006	0.003	0.0
20	9.50	0	0.0024	0.0045	0.0062	0.0073	0.0076	0.007	0.006	0.004	0.002	0.0
21	10.00	0	0.0018	0.0035	0.0048	0.0057	0.0059	0.006	0.005	0.003	0.002	0.0

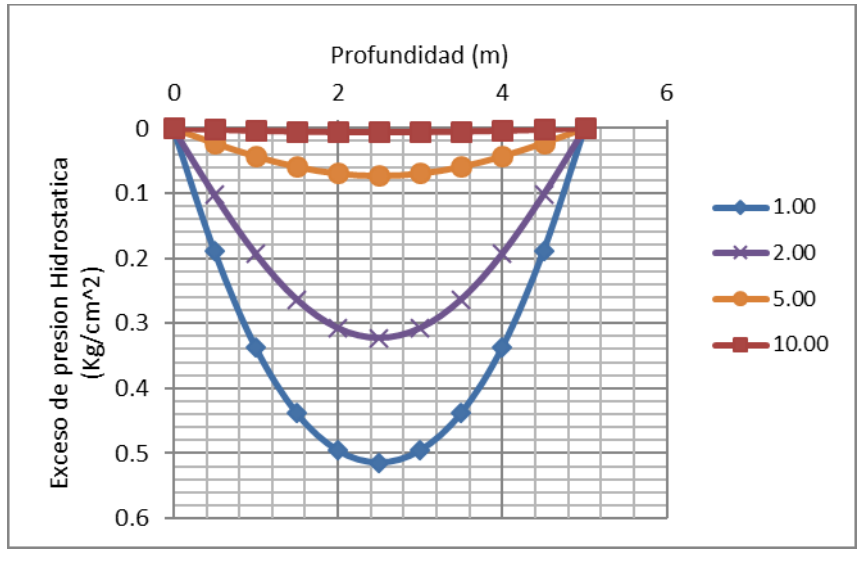


Figura 4.43.
Diagrama de exceso de presión versus profundidad de resultado Excel

Para el proceso de simulación usaremos el lenguaje de programación MATLAB. Los datos son tomados de ensayos de laboratorio de consolidación. Para la consolidación unidimensional del estrato compresible se tomará en cuenta los siguientes valores $K = 1.10^{-6}$, $dz = 0.25$, $Cv = 1.4522$, $Mv = 0.0579$ y $H = 5$. En la siguiente figura se muestra el diagrama de flujo para la solución computacional del proceso de consolidación unidimensional para un estrato compresible.

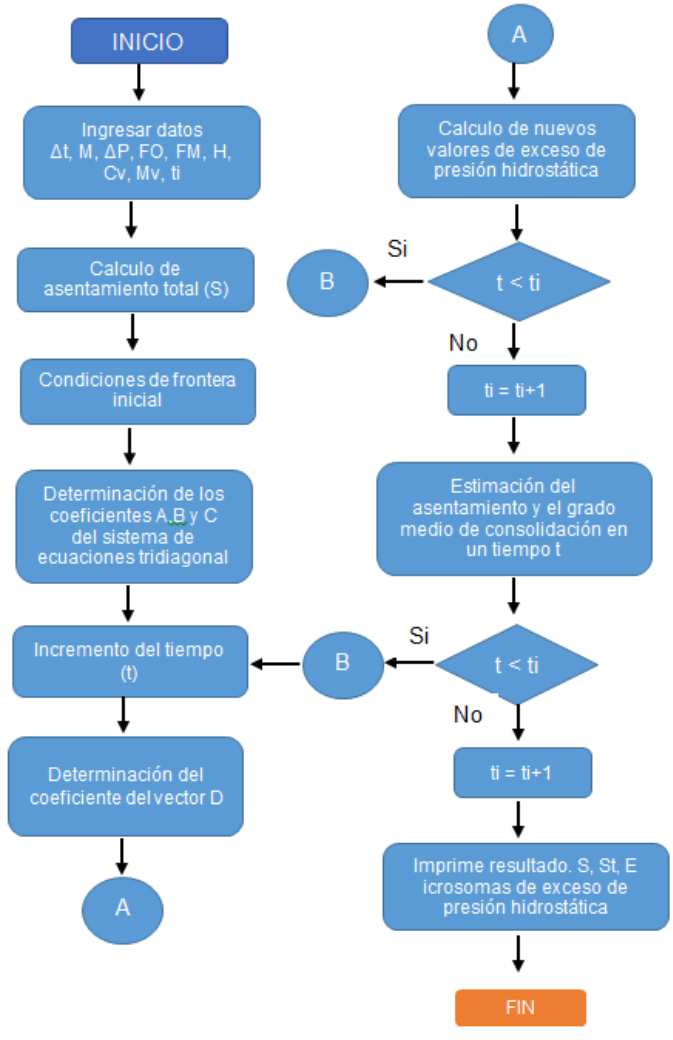


Figura 4.44.
Diagrama de flujo para la solución computacional del proceso de consolidación unidimensional para un estrato compresible

A partir del diagrama de flujo elaboramos un programa en MATLAB para la solución planteada.

```
clc; clear
T  = 5;DP =0.7;FO = 0;  FM = 0;  IT = 1;
H  = 5;    CV = 1.4522;  MV = 0.0579;   K = 1.6*10^-6;
dz = 0.25;
B  =[0:dz:H];    n=length(B);
dt = 0.1;
A  =[0:dt:T];    m=length(A);
S = MV * DP * H ;
L = CV * dt / (dz^2);
U = ones(m,n)  ;
%condiciones iniciales y de frontera

U(:,1) = FO ;
U(:,n) = FM;
U(1,:) = DP ;
U(2,:) = DP ;
 while IT<100;
    for  j=3:m
     for   i=3:n-2
     U(j,i)=((L/2)*(U(j-1,i-1)+U(j-1,i+1))+(1-L)*U(j-1,i));
end
U(j,2)= (L/2)*(U(j,1)+ U(j-1,1)+ U(j-1,3))+(1-L)*U(j-1,2);
 U(j,n-1)= (L/2)*(U(j,n)+U(j-1,n)+U(j-1,n-2))+(1-L)*U(j-1,n-1);

  BE(2)=1+L;
  GA(2)=U(j,2)/BE(2);
  for i=3:n-1
       BE(i)=(1+L)-((-L/2)*(-L/2)/BE(i-1));
       GA(i)=(U(j,i)-(-L/2)*GA(i-1))/BE(i);
  end
  U(j,n-1)=GA(n-1);
  for i=n-2:-1:2
   U(j,i)=GA(i)-((-L/2)*U(j,i+1))/BE(i);
  end
    end
IT=IT+1;
end
for j=2:m
A1= U(j,1)+U(j,n);
       for i=2:n-1
           A1=A1+2*U(j,i);
     end
        A2=(A1*dz)/2;
       AS(j)=MV*(DP*H-A2);
       E(j)=AS(j)/S;
end
```

```
U
ASENT=AS(12),AS(22),AS(51)
GMConsolidacion=E(12),E(22),E(51)
year1=U(12,:)
year2=U(22,:)
year5=U(51,:)
hold on
plot(year1)
plot(year2)
plot(year5)
grid on
```

La salida del programa elaborado reporta el siguiente resultado y las gráficas mostradas a continuación.

Tiempo	1	2	5
U	0.4265	0.6053	0.8640
Asentamiento Total	0.0864	0.1227	0.1751

Tabla 4.4.
Asentamientos totales y parciales en función del tiempo, sistema monoestrato

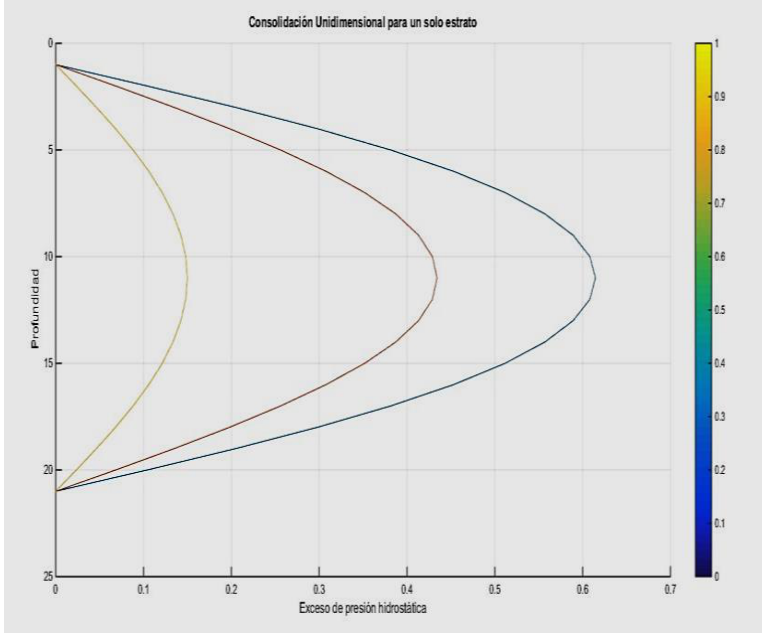


Figura 4.45.
Curva de consolidación unidimensional para un estrato

4.7.6. Consolidación Unidimensional en Suelos con Varios Estratos Compresibles

Formulación del caso:

La formulación del proceso se hará para un sistema constituido por Ns estratos compresibles (se resolverá un problema con cuatro capas deformables). La ecuación que rige el proceso de consolidación para un sistema formado por Ns estratos compresibles es:

$$\left(\frac{\partial \mu}{\partial t}\right)_L = CV_L\left(\frac{\partial^2 \mu}{\partial Z^2}\right)_L \qquad para\ L = 1, 2, \ldots, Ns \tag{4.179}$$

Donde el subíndice L denota la propiedad correspondiente al L-ésimo estrato. Las condiciones iniciales y de frontera externa que deben cumplirse para este sistema de estratos son las mismas señaladas para el caso de un solo estrato compresible. Además de estas restricciones, deben satisfacerse las condiciones de frontera interna que se presentan en las interfaces de los estratos con diferentes propiedades. Las ecuaciones correspondientes a este requisito las podemos obtener basándonos en la Ley de Darcy y en la continuidad del flujo a través de las fronteras internas, esto es:

$$\left(\frac{\partial \mu}{\partial Z}\right)_L K_L = \left(\frac{\partial \mu}{\partial Z}\right)_{L+1} K_{L+1} \quad para\ L = 1, 2, \ldots, Ns - 1 \tag{4.180}$$

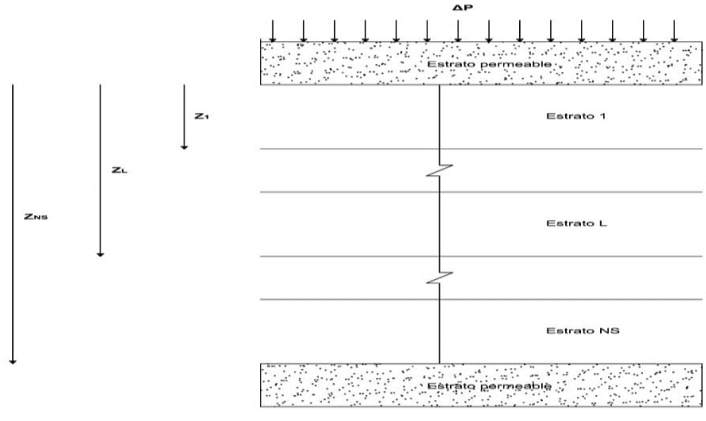


Figura 4.46.
Sistema de Ns estratos compresibles

Solución Numérica:

El modelo propuesto para este sistema de varios estratos es básicamente el mismo planteado para un estrato compresible. Aquí, al espesor total del sistema de estratos se le divide en M intervalos iguales de tamaño ΔZ; con respecto al tiempo es enteramente igual. El sistema discretizado se presenta en la siguiente figura.

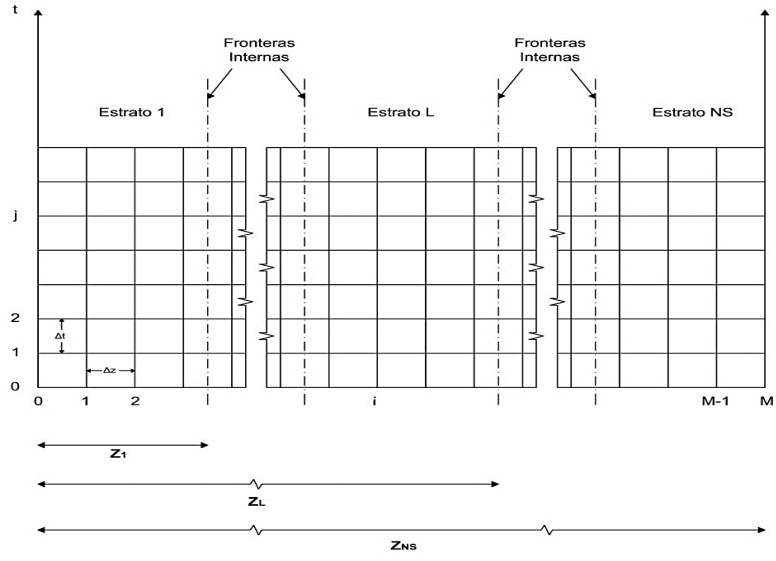


Figura 4.47.
Sistema de Ns estratos compresibles discretizados

Las expresiones discretizadas para la ecuación inicial con las condiciones iniciales y de frontera externa son las mismas que las expuestas en el caso de una capa. Esta ecuación se aplicará a todos los nodos internos de la malla, salvo aquellos puntos que se encuentren separados por una frontera interna.

Condición inicial:

$$\mu_{i,0} = f(Z) \quad para \quad i = 0,1,2,\ldots,M \tag{4.181}$$

Condiciones de frontera externa:

$$\left.\begin{matrix}\mu_{0,j} = 0 \\ \mu_{M,j} = 0\end{matrix}\right\} \quad para \quad j > 0 \tag{4.182}$$

Haciendo j=0 y aplicando la ecuación de diferencias principal a los nodos internos, con excepción de los primeros puntos situados a la izquierda, en cuyo caso deben aplicarse las siguientes ecuaciones respectivamente:

$$\begin{aligned}
&-\frac{\lambda}{2}\mu_{i-1,j+1} + \left\{1 + \frac{\lambda}{2}(2 + R_a)\right\}\mu_{i,j+1} - \frac{\lambda}{2}(1 + R_a)\mu_{i+1,j+1} \\
&= \frac{\lambda}{2}\mu_{i-1,j} + \left\{1 - \frac{\lambda}{2}(2 + R_a)\right\}\mu_{i,j} + \frac{\lambda}{2}(1 + R_a)\mu_{i+1,j} \\
&-\frac{\lambda}{2}(1 + R_b)\mu_{i-1,j+1} + \left\{1 + \frac{\lambda}{2}(2 + R_b)\right\}\mu_{i,j+1} - \frac{\lambda}{2}\mu_{i+1,j+1} \\
&= \frac{\lambda}{2}(1 + R_b)\mu_{i-1,j} + \left\{1 - \frac{\lambda}{2}(2 + R_b)\right\}\mu_{i,j} + \frac{\lambda}{2}\mu_{i+1,j}
\end{aligned} \tag{4.183}$$

Tomando en consideración las expresiones de condición inicial y de frontera externa se obtienen las ecuaciones tridiagonales de M-1 ecuaciones con M-1 incógnitas. Resolviendo este sistema de ecuaciones se pasa a la segunda línea, y así sucesivamente, de manera análoga a cómo se procedió en el caso de un solo estrato compresible. El sistema de ecuaciones que se tiene en cada línea de tiempo es de la siguiente forma:

$$b_1\mu_{1,j+1} + c_1\mu_{2,j+1} = d_1 \qquad (4.184)$$
$$a_i\mu_{i,j+1} + b_i\mu_{i,j+1} + c_i\mu_{i+1,j+1} = d_i$$
$$para \quad a \le i \le M-2$$
$$a_{M-1}\mu_{M-2,j+1} + b_{M-1}\mu_{M-1,j+1} = d_{M-1}$$

Donde para el primer renglón se tiene:

$$b_1 = (1+\lambda_1)c_1 = -\frac{\lambda_1}{2}d_1 \qquad (4.185)$$
$$= (1-\lambda_1)\mu_{i,j} + \frac{\lambda_1}{2}(\mu_{0,j+1} + \mu_{0,j} + \mu_{2,j})$$

En los nodos interiores, exceptuando los puntos con condiciones de frontera interna, se tiene:

$$a_i = -\frac{\lambda_L}{2}\, b_i = (1+\lambda_L)c_i = -\frac{\lambda_L}{2} \qquad (4.186)$$
$$d_i = (1-\lambda_L)\mu_{i,j} + \frac{\lambda}{2}(\mu_{i-1,j} + \mu_{i+1,j})$$

Para el primer nodo situado a la derecha de una frontera interna tenemos:

$$a_i = -\frac{\lambda_{L+1}}{2}(1+R_{b_L})b_i = 1 + \frac{\lambda_{L+1}}{2}(2+R_{b_L})c_i = -\frac{\lambda_{L+1}}{2} \qquad (4.187)$$
$$d_i = \frac{\lambda_{L+1}}{2}\mu_{i+1,j} + \left\{1 - \frac{\lambda_{L+1}}{2}(2+R_{b_L})\right\}\mu_{i,j} + \frac{\lambda_{L+1}}{2}(1+R_{b_L})\mu_{i-1,j}$$

Finalmente, para el último renglón del sistema será:

$$a_{M-1} = -\frac{\lambda_{N_s}}{2}\, b_{M-1} = 1 + \lambda_{N_s} d_{M-1} \qquad (4.188)$$
$$= \frac{\lambda_{N_s}}{2}(\mu_{M,j+1} + \mu_{M-1,j} + \mu_{M,j}) + (1-\lambda_{N_s})\mu_{M-1,j}$$

La solución numérica, apoyándonos en lenguaje de programación MATLAB, se formulará a partir del siguiente diagrama de flujo.

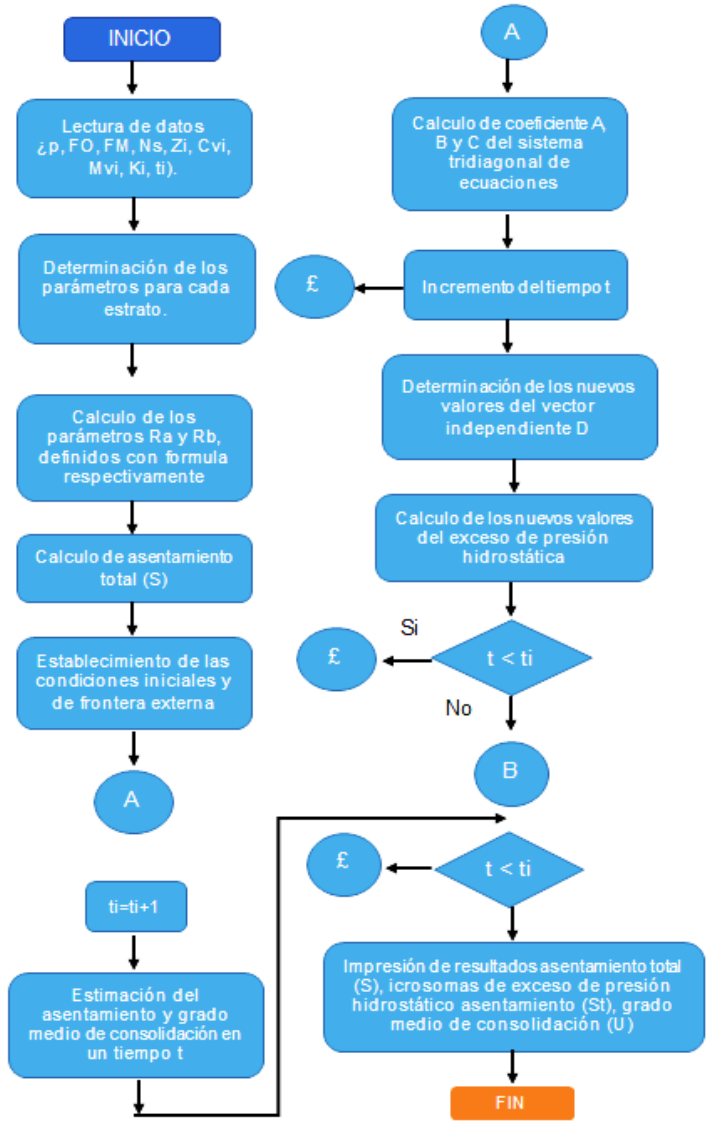


Figura 4.48.
Diagrama de flujo para la solución computacional del proceso de consolidación unidimensional con varios estratos compresibles

A partir del diagrama de flujo elaboramos un programa en MATLAB para la solución planteada con 4 estratos compresibles.

```
clc; clear
DP =1;FO = 0;FM = 0;IT = 0;NS=4;T=7195;dt=5;
CV = [0.0411 0.1918 0.0548 0.0686];
```

```
MV = [3.07*10^-3 1.95*10^-3 9.74*10^-4 1.95*10^-3] ;
K = [7.89*10^-6 2.34*10^-5 3.33*10^-6 8.35*10^-6];
H  = 80;
P=[10 30 60 80]; %profundidades
E=[10 20 30 20];% espesores
DZ =P(NS)/H;
Q  =[0:DZ:H];
n=length(Q); %numero de nodos en columnas
W  =[0:dt:T];
m=length(W);
it=0;

AT=0;I=0;
%CALCULO DE PARÁMETROS LAMBA
         for i= 1 : NS
                L(i) = CV(i) * dt / (DZ^2;
                S(i) = MV(i) * DP * E(i);
                AT = AT + S(i);
         end
         AT
         for j= 1:NS-1
                ZO = I*DZ;
                Z=P(j);
                I=I+1;
                if Z >= ZO
                    AO(j)=0.2;
                    BO(j)=(1-AO(j));
                    K1= K(j) / K(j+1);
                    K2=1/K1;
                    RA(j)=((BO(j))*(1-K1))/(K1+AO(j)*(1-K1));
                    RB(j)=((AO(j))*(1-K2))/(K2+BO(j)*(1-K2));
                end
         end

        %calculo de los coeficientes A, B, y C sistema de ecuaciones
            for w=1:4
            B(2)=1+L(1);
            C(2)=-L(1)/2;
            for i=3:P(1)
               A(i)=-L(1)/2;
               B(i)=1+L(1);
               C(i)=-L(1)/2;
            end
            A(11)=-L(1)/2;
            B(11)=1+((L(1)/2)*(2+RA(1)));
            C(11)=-L(1)/2*(1+RA(1));
            A(12)=-L(2)/2*(1+RB(1));
            B(12)=1+L(2)/2*(2+RB(1));
            C(12)=-L(2)/2;
```

```
                for i=13:P(2)
                    A(i)=-L(2)/2;
                    B(i)=1+L(2);
                    C(i)=-L(2)/2;
                end
                A(31)=-L(2)/2;
                B(31)=1+L(2)/2*(2+RA(2));
                C(31)=-L(2)/2*(1+RA(2));
                A(32)=-L(3)/2*(1+RB(2));
                B(32)=1+L(3)/2*(2+RB(2));
                C(32)=-L(3)/2;

                for i=33:P(3)
                    A(i)=-L(3)/2;
                    B(i)=1+L(3);
                    C(i)=-L(3)/2;
                end
                A(61)=-L(3)/2;
                B(61)=1+L(3)/2*(2+RA(3));
                C(61)=-L(3)/2*(1+RA(3));
                A(62)=-L(4)/2*(1+RB(3));
                B(62)=1+L(4)/2*(2+RB(3));
                C(62)=-L(4)/2;
                for i=63:P(4)-1
                    A(i)=-L(4)/2;
                    B(i)=1+L(4);
                    C(i)=-L(4)/2;
                end
                A(80)=-L(4)/2;
                B(80)=1+L(4);
                End

          U = ones(m,n)  ;
          %condiciones iniciales y de frontera
          U(:,1) = FO ;
          U(:,n) = FM ;
          U(1,:) = DP ;
          U(2,:) = DP ;

while IT < 100
            for  j=3:m
                      U(j,2)= (L(1)/2)*(U(j,1)+ U(j-1,1)+ U(j-
     1,3))+(1-L(1))*U(j-1,2); %condicion de los d1
               for   i=3:P(1)
                 U(j,i) = ((L(1)/2)*(U(j-1,i-1) + U(j-1,i+1)) +
     (1-L(1))*U(j-1,i));%estrato 1
               end
                        %frontera interna1
                        U(j,11) = ((L(1)/2)*U(j-1,10))  +  (1-
     (L(1)/2)*(2+RA(1)))*U(j-1,11)  +  (L(1)/2)*(1+RA(1))*U(j-
```

```
1,12)  ;
                    U(j,12) = ((L(2)/2)*U(j-1,13))  +  (1-
(L(2)/2)*(2+RB(1)))*U(j-1,12)  +  (L(2)/2)*(1+RB(1))*U(j-
1,11)  ;
          for   i=13:P(2)
                     U(j,i) = ((L(2)/2)*(U(j-1,i-1) + U(j-
1,i+1)) + (1-L(2))*U(j-1,i));%estrato 2
          end
                     %frontera interna2
                    U(j,31) = ((L(2)/2)*U(j-1,30))  +  (1-
(L(2)/2)*(2+RA(2)))*U(j-1,31)  +  (L(2)/2)*(1+RA(2))*U(j-
1,32)  ;
                    U(j,32) = ((L(3)/2)*U(j-1,33))  +  (1-
(L(3)/2)*(2+RB(2)))*U(j-1,32)  +  (L(3)/2)*(1+RB(2))*U(j-
1,31)  ;
          for   i=33:P(3)
                     U(j,i) = ((L(3)/2)*(U(j-1,i-1) + U(j-
1,i+1)) + (1-L(3))*U(j-1,i));%estrato 3
          end
                    %frontera interna3
                    U(j,61) = ((L(3)/2)*U(j-1,60))  +  (1-
(L(3)/2)*(2+RA(3)))*U(j-1,61)  +  (L(3)/2)*(1+RA(3))*U(j-
1,62)  ;
                    U(j,62) = ((L(4)/2)*U(j-1,63))  +  (1-
(L(4)/2)*(2+RB(3)))*U(j-1,62)  +  (L(4)/2)*(1+RB(3))*U(j-
1,61)  ;
          for   i=63:(P(4)-1)
                     U(j,i) = ((L(4)/2)*(U(j-1,i-1) + U(j-
1,i+1)) + (1-L(4))*U(j-1,i));%estrato 4
          end
             U(j,n-1)= (L(4)/2)*(U(j,n)+U(j-1,n)+U(j-1,n-
2))+(1-L(4))*U(j-1,n-1);
       BE(2)=B(2);
       GA(2)=U(j,2)/BE(2);
           for i=3:n-1
           BE(i)=B(i)-((A(i)*C(i-1))/BE(i-1));
           GA(i)=(U(j,i)-(A(i)*GA(i-1)))/BE(i);
           end
           U(j,n-1)=GA(n-1);
           for i=n-2:-1:2
           U(j,i)=(GA(i)-(C(i)*U(j,i+1))/BE(i));
           end
    end
 IT=IT+1;
end
JO=1;
ASE=0;
for j=1:NS
          JI=P(j);
          A1= U(1,JO+1)+U(m,JI+1);
          for i=JO+1:JI+1
```

```
            A1=A1+2*U(j,i);
    end
    A2=(A1*DZ)/2;
    AS(j)=DP*MV(j)*P(j)-(MV(j)*A2);

    ASE=ASE+AS(j);
    JO=P(j);
end
ASE;
ES=(ASE)/AT;
U;

time1=U(148,:)
time2=U(587,:)
time3=U(m,:)
hold on
plot(time1)
plot(time2)
plot(time3)
grid on
```

La salida del programa elaborado reporta el siguiente resultado y gráficas.

Tiempo	1	2	5
U	0.756	0.506	0.252
Tiempo (días)	7 195	2 930	740

Tabla 4.5.
Asentamientos totales y parciales en función del tiempo, sistema multiestrato

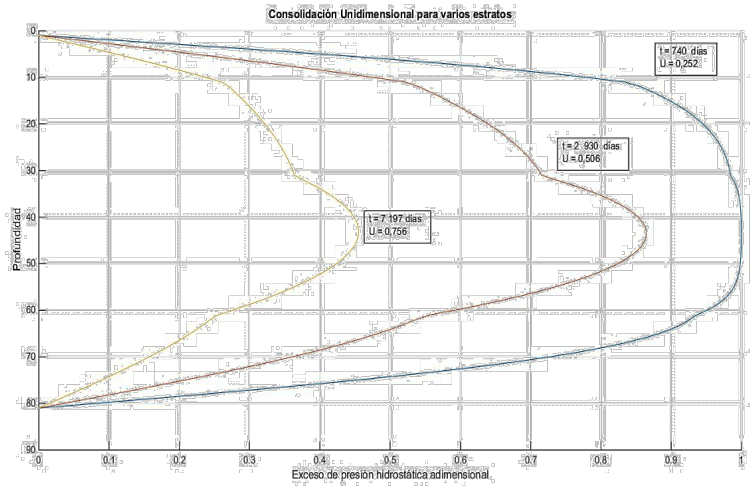


Figura 4.49.
Curva de consolidación unidimensional para 4 estratos compresibles

4.8. Problemas Propuestos

- Problema propuesto 1: La ecuación diferencial parcial del tipo parabólico que se muestra, modela el flujo o propagación de calor en una barra homogénea de longitud finita sin fuentes de calor.

$$\frac{\partial u}{\partial t} = \frac{k}{\rho C_p}\frac{\partial^2 u}{\partial x^2}, \quad \text{en} \quad 0 < x < L, \; t > 0$$
$$u(0,t) = u(L,t) = 0, \quad t > 0,$$
$$u(x,0) = u^0(x), \quad 0 \le x \le L,$$

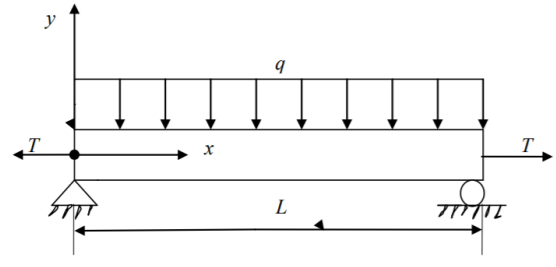
Donde $u = u(x, t)$ es la temperatura de la barra en la posición x al tiempo t; k es el coeficiente de conductividad térmica; ρ es la densidad de la barra; Cp es el calor específico, y L es la longitud de la barra. Suponemos que los parámetros k, ρ y Cp son constantes. Se pide establecer la ecuación en diferencias finitas bajo un esquema de solución apropiado, justifique su respuesta.

- Problema propuesto 2: Usando la ecuación de Laplace, obtener la distribución de temperatura en estado estacionario para una placa calentada según la siguiente información:

$$\nabla^2 u(x,y) = \frac{\partial^2 u(x,y)}{\partial x^2} + \frac{\partial^2 u(x,y)}{\partial y^2} = 0, para\ 0 \le x \le 4, 0 \le y \le 4$$
con las siguientes condiciones de frontera:
$$u(0,y) = e^y - \cos(y) \ldots\ldots\ldots u(4,y) = e^y \cos(4) - e^4 \cos(y)$$
$$u(x,0) = \cos(x) - e^x \ldots\ldots\ldots u(x,4) = e^4 \cos(x) - e^x \cos(4)$$

- Problema propuesto 3: La flexión en una viga, simplemente apoyada con una carga uniforme q y carga axial de tracción T, viene dado por:

$$\frac{d^2 y}{dx^2} - \frac{Ty}{EI} = \frac{qx(L-x)}{2EI}$$



Donde:
x = Posición a lo largo de la viga (in)
T = Tensión aplicada (lbs)
E = Módulo de elasticidad de la viga (psi)
I = Segundo momento de inercia (in^4)
q = Carga uniformemente distribuida (lb/in)
L = Longitud de la viga (in)

Dado: $T = 7200$ lbs; $q = 5400$ lbs/in; $L = 75$ in; $E = 30$ Msi y $I = 120$ in^4.

(a) Encuentre la deflexión de la viga en $x = 50$ in. Use un tamaño de paso de Δx=25 in y aproxime las derivadas por aproximación de diferencia central dividida.
(b) Encuentre el error relativo verdadero en el cálculo de $y(50)$.

- Problema propuesto 4: Tomemos el caso de un recipiente a presión que se está probando en el laboratorio para verificar su capacidad para soportar presión. El recipiente tiene radio interno a y radio externo b. La ecuación diferencial para el desplazamiento radial u de un punto a lo largo del grosor viene dada por:

$$\frac{d^2u}{dr^2} + \frac{1}{r}\frac{du}{dr} - \frac{u}{r^2} = 0$$

Asimismo, el radio interno a = 5 in y el radio externo b = 8 in y el material del recipiente a presión es ASTM A36 acero. El límite elástico de este tipo de acero es de 36 ksi. Dos galgas extensiométricas que son unidas tangencialmente en el radio interno y externo miden la tensión tangencial normal reportando a la presión máxima necesaria:

$$\epsilon_{t/r=a} = 0.00077462$$
$$\epsilon_{t/r=b} = 0.00038462$$

Dado que el desplazamiento radial y la tensión tangencial están relacionados por:

$$\epsilon_t = \frac{u}{r},$$

Entonces:

$$u\big|_{r=a} = 0.00077462 \times 5 = 0.0038731''$$
$$u\big|_{r=b} = 0.00038462 \times 8 = 0.0030769''$$

El esfuerzo normal máximo en el recipiente a presión está en el radio interno $r = a$ y está dado por:

$$\sigma_{\max} = \frac{E}{1-\nu^2}\left(\frac{u}{r}\bigg|_{r=a} + \nu\frac{du}{dr}\bigg|_{r=a}\right)$$

Donde:
E = Módulo de Young para el acero (E = 30 Msi)
ν = relación de Poisson (ν = 0.3)

El factor de seguridad (FS) está dado por FS=Límite elástico del acero/σ_{max}

Se pide:

(a) Dividir el espesor radial del recipiente a presión en 6 nodos equidistantes y encuentre el perfil de desplazamiento radial.

(b) Encuentre el esfuerzo normal máximo y el factor de seguridad.

(c) Encuentre el valor exacto del esfuerzo normal máximo dado por la ecuación para FS si la expresión exacta para desplazamiento radial es de la forma:

$$u = C_1 r + \frac{C_2}{r}$$

CAPÍTULO V
APLICACIÓN PDETOOL DE MATLAB

5.1. Sobre PDETool de MATLAB

La caja de herramientas de MATLAB ("Toolbox"), nos permite resolver ecuaciones diferenciales en derivadas parciales utilizando el método de elementos finitos. En particular, con la utilización de la interfaz gráfica denominada "PDETool" es posible especificar el dominio 2D de un problema, la triangulación del dominio, los coeficientes de la EDP y las condiciones de contorno.

El método tradicional de solución numérica toma los siguientes pasos: descripción de la geometría, generación de la malla de elementos, discretización de la ecuación, condiciones de borde o frontera, solución del sistema de ecuaciones y gráfica de resultados.

La tecnología computacional actual ha desarrollado programas orientados a la solución de estos problemas considerando la automatización de los pasos anteriores. Aunque matemáticamente las ecuaciones son similares, los programas están clasificados considerando la aplicación final. Así tenemos programas para análisis de esfuerzos mecánicos, fluido dinámico computacional (CFD), sistemas electromagnéticos y genéricos para uso general. En el área de ingeniería de procesos son usados los de tipo CFD y de configuración general para resolver problemas de fluido-dinámica, transferencia de calor, transferencia de masa, reactores, etc.

Por tanto, PDETool de MATLAB es una herramienta que facilita la resolución de problemas de EDP. La solución presentada se obtiene haciendo uso del método de elementos finitos para problemas sobre dominios limitados y continuos en el plano. Los casos incluidos son: transferencia de calor en estado estable y transitorio, flujo en medios porosos y problemas de difusión, propagación de ondas transitorias y armónicas, movimientos transversales en membranas, determinación de estados de vibración natural de membranas y problemas de estructuras mecánicas.

5.2. Cómo Utilizar PDETool de MATLAB

En resumen, presentamos algunos de los elementos básicos para trabajar con el "Toolbox", para lo cual debemos comenzar a trabajar con la interfaz gráfica para EDP de MATLAB tecleando PDETool en la línea de comandos.

5.2.1. Estableciendo Dominios

Para especificar el dominio de nuestro problema procederemos del siguiente modo: en "Options", elegir "Axes Limits" para concretar el área (x, y) (por defecto, $x = -1.5 : 1.5$ e $y = -1 : 1$); seleccionar "Grid" para situar un mallado sobre el área que aparece y "Snap" para ajustar automáticamente la figura de nuestro dominio al mallado (esto facilita dibujar regiones simples).

Seleccionar "Grid Spacing" si se desea modificar el espaciado del mallado (el que está por defecto es 0.5 en x y 0.2 en y). Los dominios se construyen en base a la composición (suma y/o resta) de una serie de dominios elementales tales como rectángulos, polígonos y elipses (los que aparecen en el menú de opciones), cuya manipulación es muy sencilla. Se recomienda escoger un par de ejemplos para comprobarlo.

Para formar el área de interés, escribiremos la "fórmula" que describe nuestro dominio como suma y/o resta de las figuras elementales en el espacio destinado a "Set formula" (que aparece sobre la región gráfica). Si queremos guardar este dominio para utilizarlo posteriormente, seleccionaremos "Export geometry description" del menú de la opción "Draw" y elegiremos "OK" en el cuadro mostrado. De este modo "exportaremos" los datos de la geometría bajo los nombres de las variables "gd" "sf" "ns" (salvo que se cambie). De hecho, sin entrar en detalles, esta especificación de la geometría debe ser procesada antes de poder utilizarla con diferentes comandos MATLAB. Es más sencillo esperar a tener especificadas las condiciones de contorno y exportar simultáneamente estas y la geometría del problema.

5.2.2. Condiciones de Contorno

Para establecer las condiciones de contorno seleccionar en "Boundary" la opción "Boundary mode" o, equivalentemente, pinchar el botón "$\partial\Omega$". Seleccionar uno o varios segmentos y en el menú de "Boundary" elegir "Specify Boundary Conditions". Se pueden especificar condiciones de contorno tipo Dirichlet de la forma $h*u=r$ introduciendo los valores de los parámetros h y r (por defecto, $h = 1$ y $r = 0$). Si elegimos condiciones de contorno tipo Neumann, deberemos especificar los coeficientes g y q. La forma general de la condición de contorno aparece en la parte superior del cuadro de diálogo (n es el vector unitario normal y c es un coeficiente en la EDP). Para guardar las condiciones de contorno en una forma que permita ser utilizada en un programa MATLAB, seleccionar "Export Decomposed Geometry", "Boundary Cond's" en el menú de "Bundary" y elegir "OK" en el cuadro. De este modo exportaremos los datos de geometría y condiciones de contorno.

5.2.3. Especificaciones de la EDP a Resolver

Para especificar la EDP a resolver, seleccionaremos "PDE specification" en el menú de la opción "PDE" y escogeremos un problema prototipo (por defecto, elíptico, de la forma -div(c*grad(u))+ $a*u = f$). Introducir los coeficientes "c" y "a" de la EDP y la función "f" del lado derecho de la ecuación. Los coeficientes pueden ser funciones de "x", "y" y para problemas no lineales también pueden ser función de "u", "ux", "uy". Utilizaremos entonces "x", "y", "u", "ux", "uy" en la

expresión a introducir, dándonos cuenta de que MATLAB interpreta estas cantidades como vectores. Teniendo en cuenta esto si queremos introducir la función "xy", teclearemos "$x.*y$" en lugar de "$x*y$".

5.2.4. Selección de Parámetros

Seleccionando "Parameters" en el menú de la opción "Mesh" estableceremos (opcionalmente) los parámetros de la triangulación inicial de nuestro problema de elementos finitos. Para generar la triangulación pincharemos en el botón con el triángulo o, alternativamente, elegiremos "Initialize Mesh" en el menú de "Mesh". Para guardar la triangulación de forma que pueda ser utilizada en un programa MATLAB, seleccionaremos "Export Mesh" en el menú de "Mesh" y pincharemos "OK" en el cuadro de diálogo. Esto permite exportar los vértices, los bordes y la ordenación de los triángulos con los nombres: "p" "e" "t" (salvo cambio), respectivamente. Démonos cuenta de que necesitaremos esta información si queremos comparar la solución exacta y la aproximada en los vértices o calcular diversas formas del error.

5.2.5. Refinando la Malla

Para refinar la triangulación pincharemos el triángulo dividido o seleccionaremos "Refine Mesh" en el menú de "Mesh". El método de refinamiento por defecto es regular (para cambiarlo iremos a "Parameters"). La solución numérica de la EDP puede realizarse de manera adaptativa seleccionando primero "Solve Parameters" y luego "Adaptive Mode" en el menú de la opción "Solve": introduciremos el máximo número de triángulos que deseamos o el máximo número de pasos de refinamiento.

5.2.6. Solución de la EDP

Para resolver la EDP y dibujar la solución seleccionaremos "Solve PDE" en el menú de "Solve". Para exportar la solución al espacio de trabajo principal de MATLAB elegiremos "Export Solution". Disponemos de varias opciones de gráficos para representar la solución; seleccionaremos "Parameters" del menú de "Plot".

5.3. Estudio de Casos

5.3.1. Líneas de Flujo o Corriente y Equipotenciales

Se puede estudiar el flujo de líneas de corriente en diferentes situaciones; por ejemplo, la forma más sencilla corresponde al trazo de líneas de flujo uniforme. Establecemos condiciones de borde tipo Neumann en la parte superior e inferior (es decir intercambio nulo) y condiciones de borde tipo Dirichlet en el borde izquierdo (20 unidades) y derecho (4 unidades) obteniéndose lo siguiente: se debe establecer "PDE Specification" de la siguiente manera con f=0.

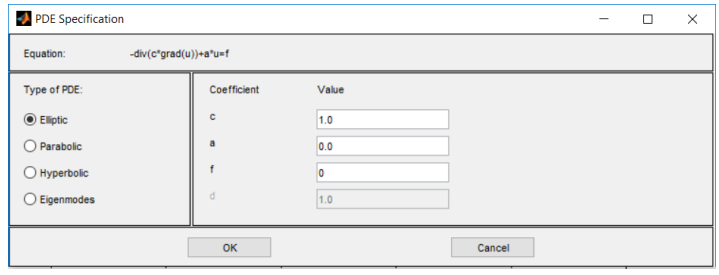


Figura 5.1.
Especificaciones para EDP

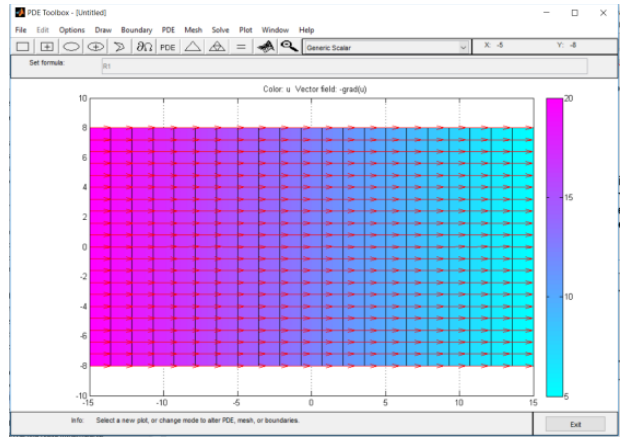


Figura 5.2.
Flujo uniforme de izquierda a derecha

A este flujo uniforme podemos colocarle un obstáculo, quizá simulando la base del pilar de un puente, en este caso un pilar de sección rectangular o cuadrada (condición de frontera tipo Neumann).

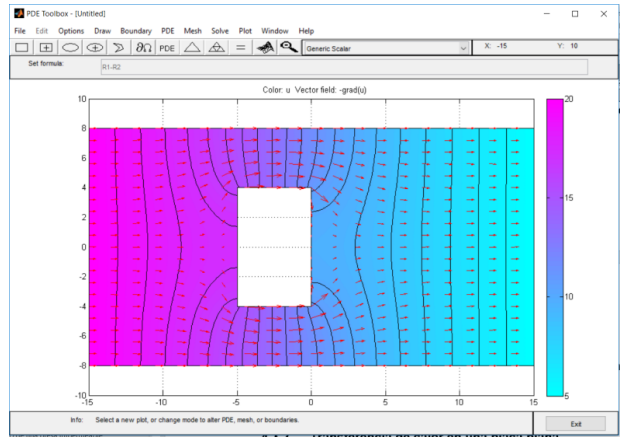


Figura 5.3.
Flujo uniforme de izquierda a derecha incluyendo un obstáculo de sección rectangular

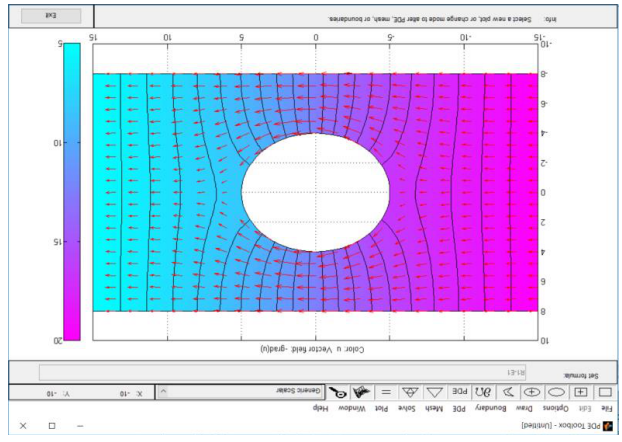


Figura 5.4.
Flujo uniforme de izquierda a derecha incluyendo un obstáculo de sección circular

También es posible realizar combinaciones de flujo; por ejemplo, un flujo uniforme de izquierda a derecha más una fuente y un sumidero. La fuente o sumidero serán condiciones tipo Newman, positiva si es fuente, negativa si es sumidero.

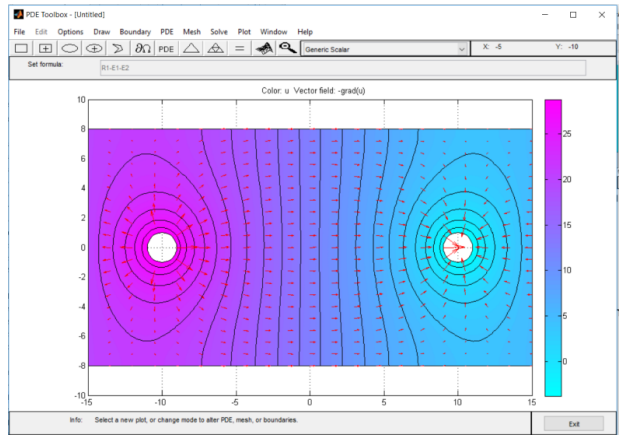


Figura 5.5.
Flujo uniforme de izquierda a derecha incluyendo una fuente y un sumidero

5.3.2. Transferencia de Calor en una Placa Plana

Se estudia mediante el método de elementos finitos el comportamiento de la conducción de calor de una placa plana, tal como se muestra en la figura:

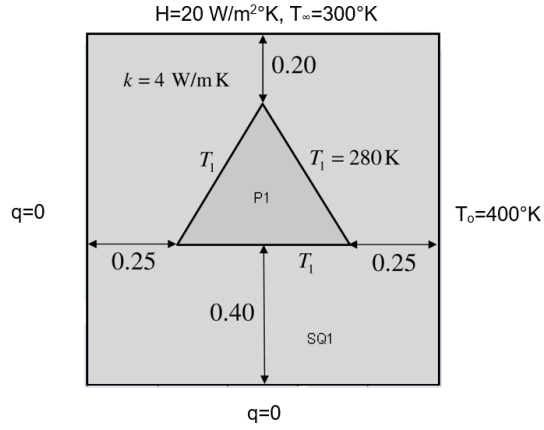


Figura 5.6.
Placa plana para transferencia de calor

La placa es de longitud "l x l", área cuadrada con orificio triangular, condiciones convectivas y adiabáticas a temperatura "T" constante. Cargamos PDETool y luego seleccionamos "Heat Transfer" e iniciamos a construir la geometría de la placa en el área de trabajo.

a) Transferencia de calor estacionaria

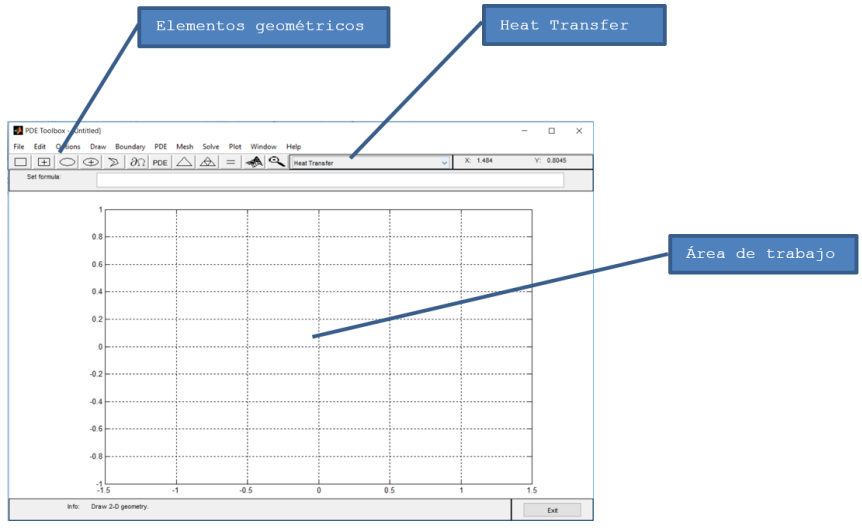


Figura 5.7.
Espacio de trabajo, elementos geométricos

Utilizamos las siguientes opciones:
'Options' 'Axes Equal'
'Options' 'Axes Limits'… set -0.5 to 1.5 for x and y.
'Options' 'Grid'
Graficar el rectángulo y el triángulo interior según la geometría del problema.

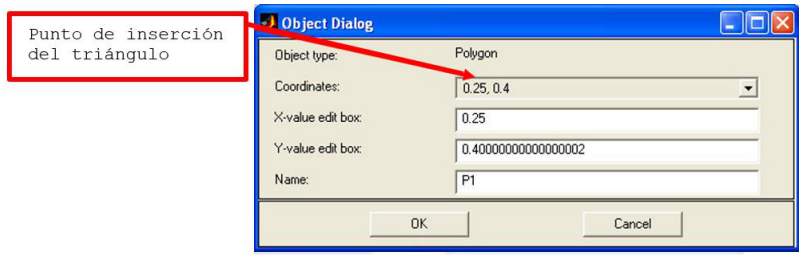


Figura 5.8.
Caja de diálogo con punto de inserción

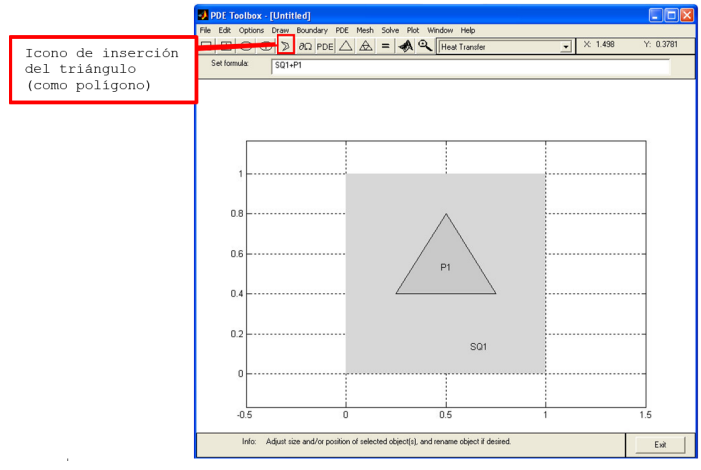


Figura 5.9.
Inserción de figuras geométricas, triángulo

Se definió el cuadrado $SQ1$ y el triángulo $P1$. El dominio es $SQ1$-$P1$.

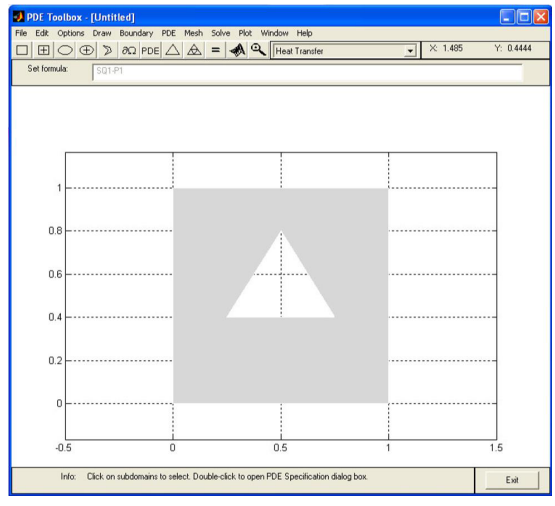


Figura 5.10.
Generando placa con agujero triangular

A continuación, definimos las condiciones de borde para la placa cuadrada:
Lado derecho: Dirichlet.

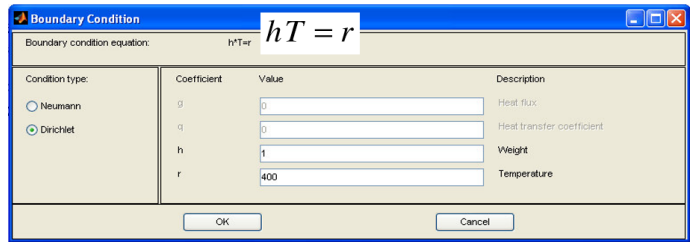


Figura 5.11.
Condiciones de borde tipo Dirichlet

Lados inferior e izquierdo: Neumann.

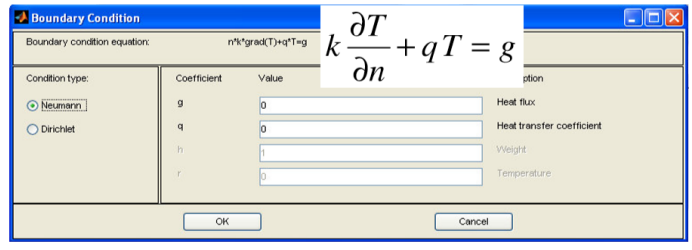


Figura 5.12.
Condiciones de borde tipo Neumann

Lado superior: Condición de borde convectiva (Neumann):

$$flux = -k\frac{\partial T}{\partial n} = h(T - T_\infty)$$

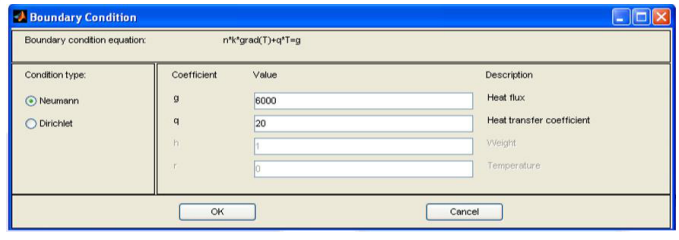


Figura 5.13.
Condiciones de borde tipo Neumann-convectiva

Condiciones de borde o frontera para el triángulo interior; los tres lados corresponden a condiciones tipo Dirichlet con T=280 °K.

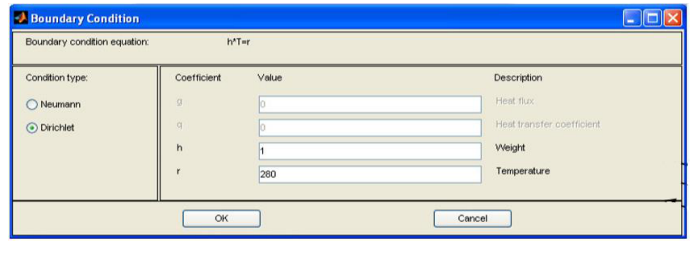


Figura 5.14.
Condiciones de borde Dirichlet de T° constante

En el menú 'PDE' seleccionar 'PDE Mode' y luego hacer doble clic en el dominio.

$$-\nabla.(k\nabla T) = Q + h(T_{ext} - T), \ldots\ldots en\ esta\ ecuación\ Q = 0; h = 0$$

En esta forma de "PDE", el término de convección y calor que aparece son relativos a todo el dominio y no a las condiciones de borde.

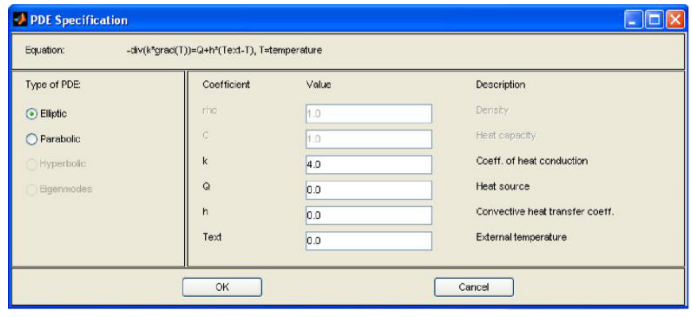


Figura 5.15.
Caja de diálogo para especificar EDP

En el menú 'Mesh' seleccionar 'Mesh Mode' y en el mismo menú afinar con 'Refine Mesh'.

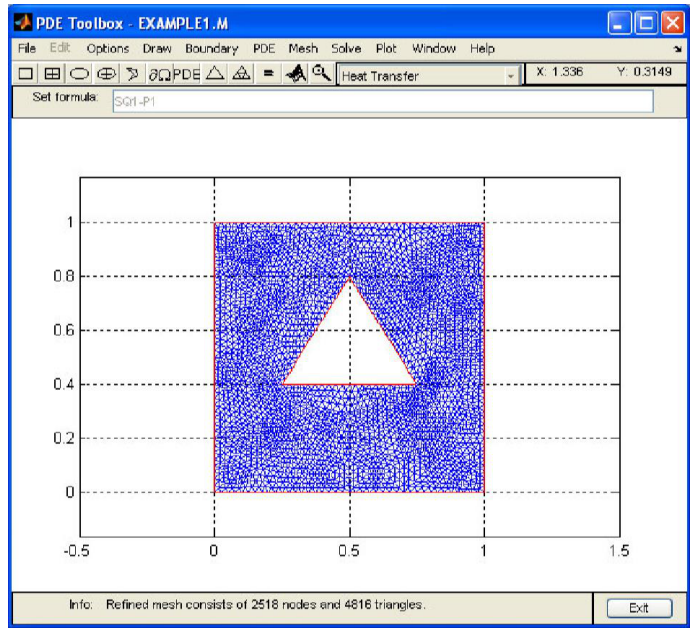


Figura 5.16.
Refinado de malla

En el menú 'Plot' seleccionar 'Parameters' y especificar las opciones de visualización. Finalmente, activar 'Plot'.

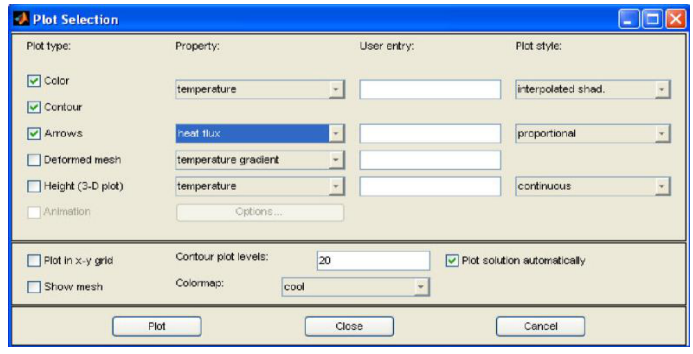


Figura 5.17.
Caja de diálogo para definición de parámetros

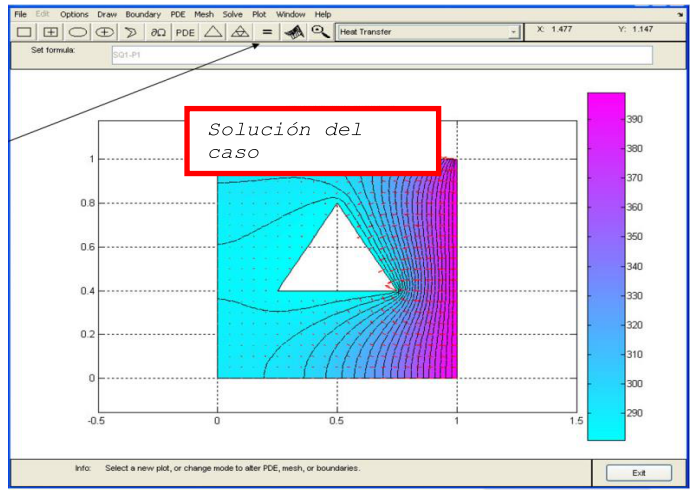


Figura 5.18.
Simulación, resultados en 2D

Si activamos 'height (3D plot)', gráfica en 3D con plano de contorno, obtendremos:

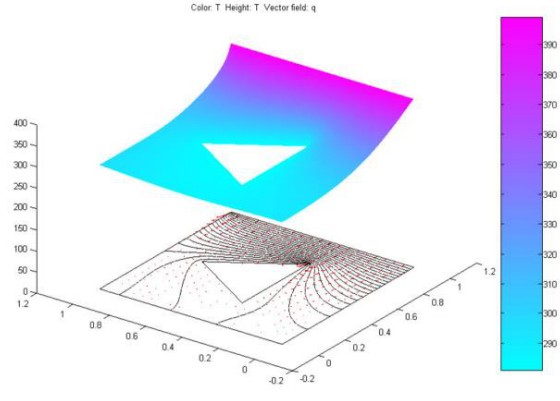


Figura 5.19.
Simulación, resultados en 3D

b) Transferencia de calor no estacionaria.

En este caso usaremos la ecuación completa:

$$-\nabla.(k\nabla T) = Q + h(T_{ext} - T)$$

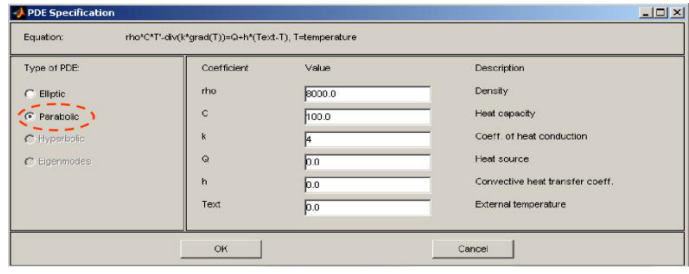


Figura 5.20.
Caja de diálogo para especificar EDP Parabólica

Los parámetros de simulación son: En el menú 'Solve' seleccionar 'Parameters'

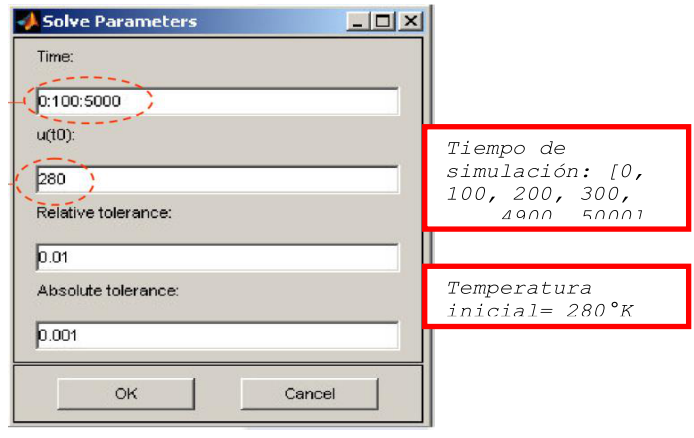


Figura 5.21.
Caja de diálogo para especificar tiempos de simulación y valores iniciales

Finalmente, en el menú 'Plot' seleccionar 'Parameters' y luego 'Animation' para animar la solución con el tiempo.

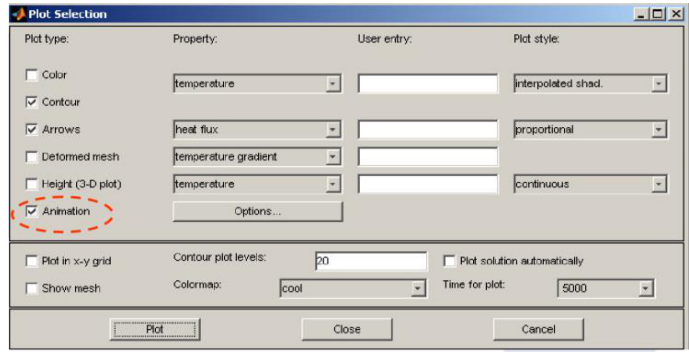


Figura 5.22.
Parámetros para simulación de estado transitorio

5.3.3. Viga en Voladizo Sujeta a Carga Puntual

Resolveremos el problema de elasticidad en 2D como se muestra a continuación:

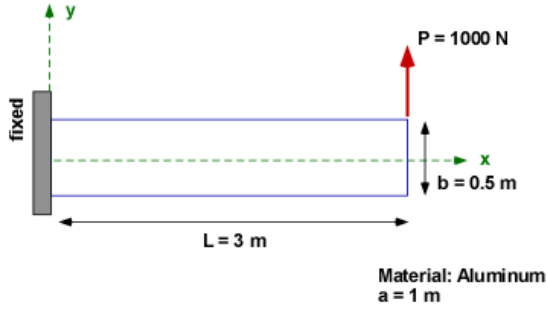


Figura 5.23.
Viga en voladizo

Iniciamos configurando el caso de la siguiente manera, usando como tipo de problema 'Structural Mech., Plane Stress':

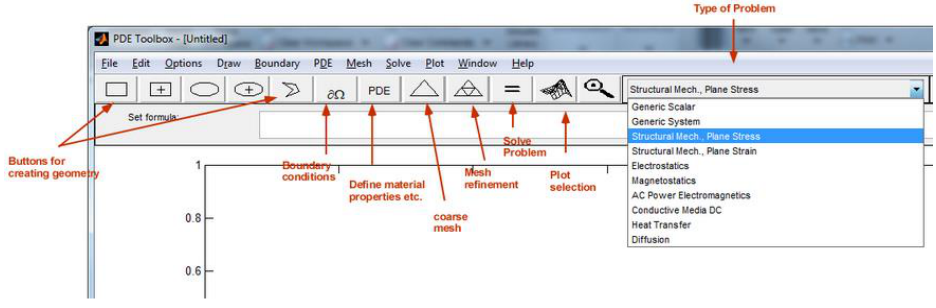


Figura 5.24.
Definición de tipo y geometría para viga en voladizo

A continuación, vamos a configurar el problema; recordemos que se está resolviendo para los desplazamientos, pero se puede obtener la tensión y los esfuerzos de esta solución.

- Paso 1: Use menú para definir la región de trabajo:
Options → Axes Limits (un espacio mayor que el rectángulo de la viga)
Set X-axis rango [-0.5 3.5]
Set Y-axis rango [-0.5 1.0]
Options → Grid (Draw grid)
Options → Snap
"Click Rectangle icon"
Use el mouse para dibujar el rectángulo
Clic en coordenadas (0,0.5) presiones y arrastre hasta coordenadas (3,0)

- Paso 2: Condiciones de borde:
Boundary → eliminar todos los límites del subdominio
"Click boundary conditions button"

(i) Borde izquierdo: Doble clic en el borde izquierdo ($x = 0$). No hay desplazamiento ($u = v = 0$). Recuerde que en la figura siguiente es el vector de u y v. Usaremos la condición de borde tipo Dirichlet.

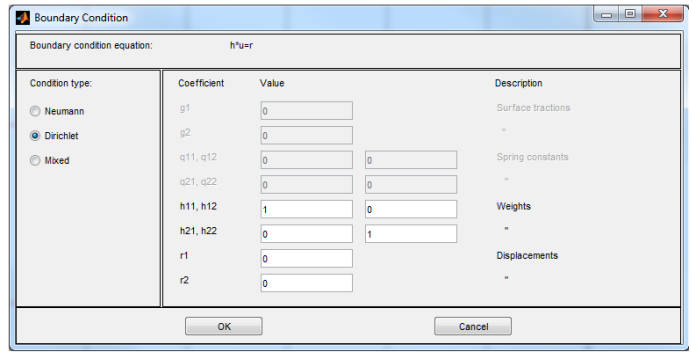


Figura 5.25.
Condiciones de borde de viga en voladizo (1)

(ii) Borde derecho: Esto lleva la carga "P". No puede configurar una carga puntual. Debe especificarlo como una fuerza de tracción o Fuerza / Área en el borde. Eso es $P / b = 2000\ \mathrm{N}$. Tenemos que especificar la tracción usando la condición de contorno de Neumann. Haga doble clic en el borde derecho ($x = 3$)

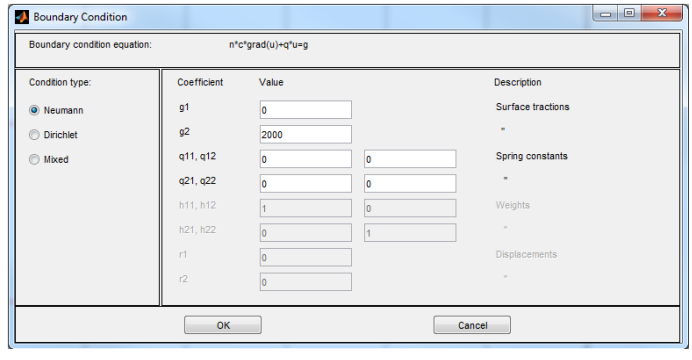


Figura 5.26.
Condiciones de borde de viga en voladizo (2)

(iii) Borde superior ($y = 0.5$): Toda la tracción es cero, usaremos condición de borde tipo Neumann para aceptar valores predeterminados.

(iv) Borde inferior: Lo mismo que para el borde superior.

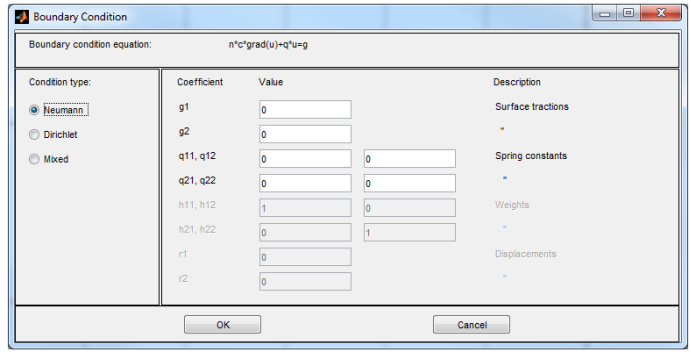


Figura 5.27.
Condiciones de borde de viga en voladizo (3)

- Paso 3: Hacer clic en PDE botón para establecer las propiedades del material para Aluminio:
density=2710 $\mathrm{kg/m^3}$
Modulus of Elasticity (E) = 70 Gpa
Modulus of rigidity (G) = 26 Gpa
Poisson ratio (nu): calcular a partir de $G = E/[2(1+nu)]$

- Paso 4: Malla del dominio (default meshes):
Haga clic en malla gruesa, refina la malla dos veces. Este programa solo usa mallas o elementos triangulares.

- Paso 5: (Solución del problema):
Clic en icono "="
Veremos la solución, se puede cambiar las propiedades de trazado haciendo clic en el botón de trazado.

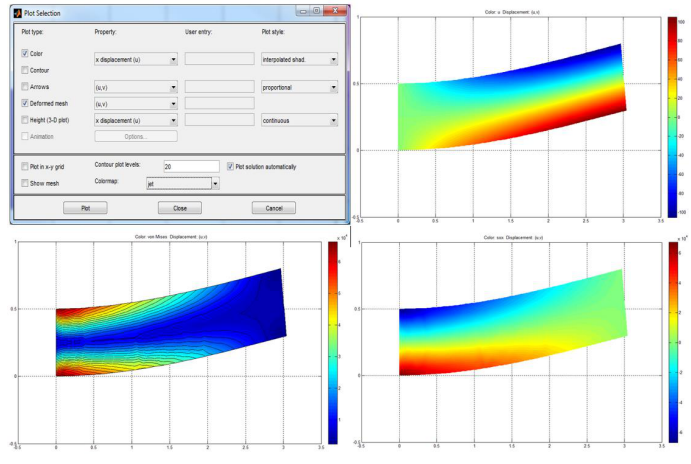


Figura 5.28.
Parámetros y resultados de viga en voladizo

5.3.4. Flujo Debajo de una Presa Impermeable e Incorporación de un Dren

Para el estudio del flujo de agua debajo de una presa de material impermeable usaremos el caso de la siguiente figura. Se muestra las dimensiones del problema, se tiene un solo estrato identificado como Suelo 1, es isotrópico, es decir que la permeabilidad $K_x = k_y = 10^{-5}$ m/s. En la parte central de la base lleva una pantalla impermeable de 10 m de profundidad por 1 m de espesor. La base del cuerpo de presa es de 10 m y se encuentra cimentada a 1 m de profundidad.

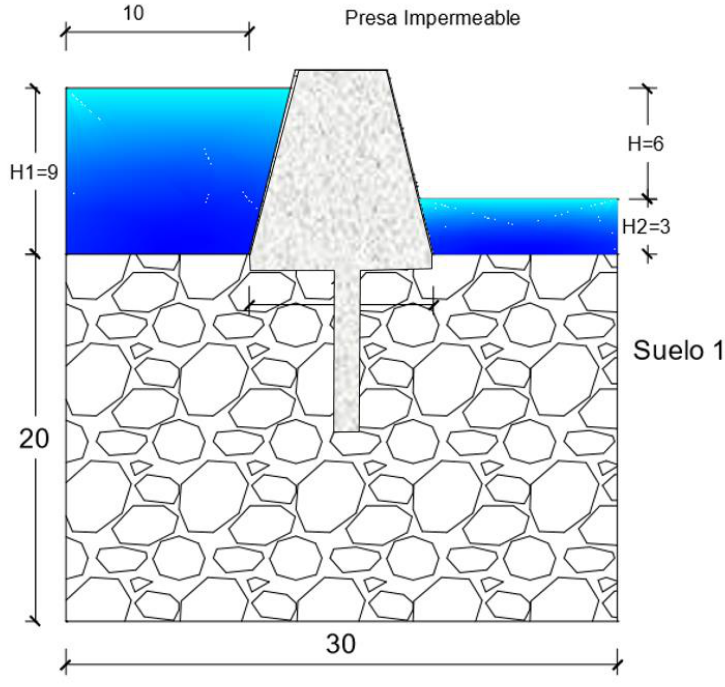


Figura 5.29.
Presa impermeable con pantalla

Como en los casos anteriores, procederemos a activar PDETool en la línea de comando de MATLAB. Elegiremos el tipo "Generic Scalar". Como primer paso definiremos la geometría del sistema, luego las condiciones de borde o frontera especificando bordes tipo Neumann y Dirichlet.

Además de los ejes iguales, los "grid" y los límites, incluimos "X-axis extra ticks", valores -0.5 y 0.5, de igual manera en "Y-axis extra ticks", el valor -7, con la finalidad de poder trazar con facilidad la pantalla impermeable.

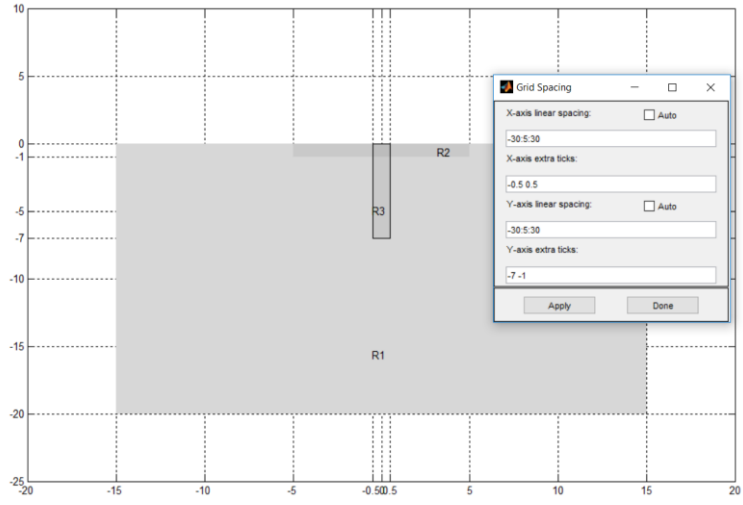


Figura 5.30.
Geometría de la presa

Luego generamos las regiones de la siguiente manera: R1-R2-R3 para paso seguido refinar la malla.

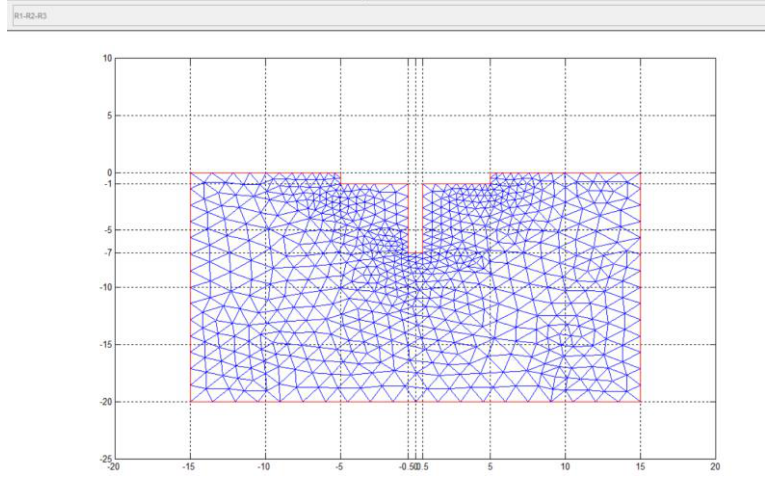


Figura 5.31.
Refinado de la malla

A continuación, establecemos las condiciones de borde tipo Neuman; se asignará a la base de la presa y toda la pantalla impermeable, así como a los límites del modelo izquierdo, superior e inferior. La parte de contacto con el agua aguas arriba será una frontera tipo Dirichlet con 9 m de carga y la parte de aguas abajo con 3 m de carga.

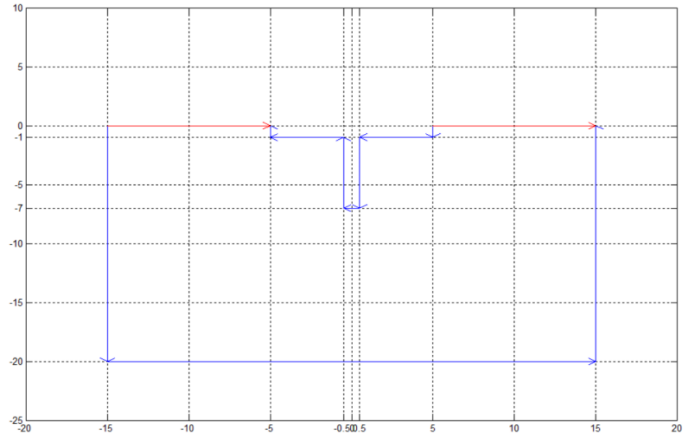


Figura 5.32.
Condiciones de borde

Revisamos la EDP para verificar la condición:

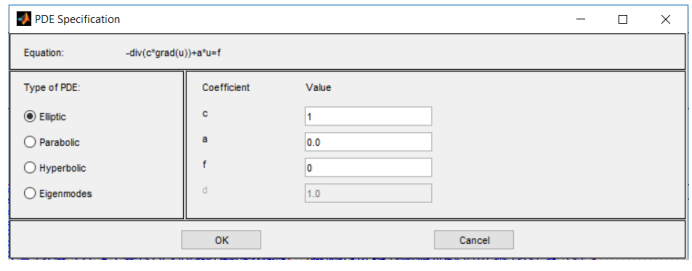


Figura 5.33.
Revisión de "PDE Specification"

Finalmente, ejecutamos la simulación, obteniéndose el siguiente resultado, que representa las líneas de flujo y las líneas de potencial correspondientes al flujo debajo de una presa impermeable con pantalla:

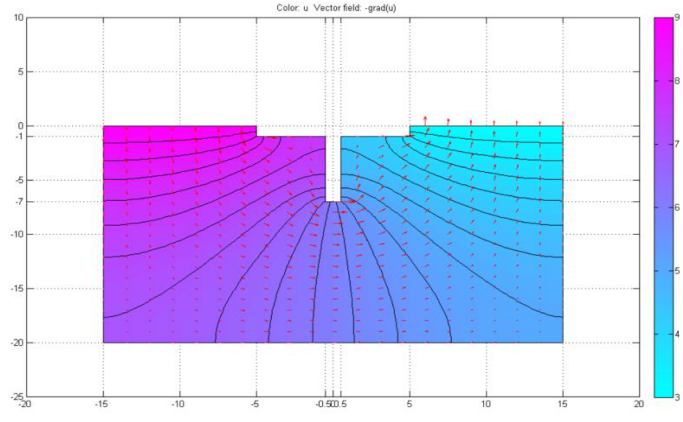


Figura 5.34.
Resultados de simulación de flujo bajo la presa impermeable

En el siguiente paso, incluimos un dren en la parte inferior de la pantalla de la presa. El caudal asignado al dren corresponde a 20 unidades cúbicas/tiempo. En este caso debemos notar que el círculo asignado al dren se divide en 4 arcos iguales, por lo tanto, cada arco representa 5 unidades cúbicas/tiempo y con signo negativo que va a generar una extracción.

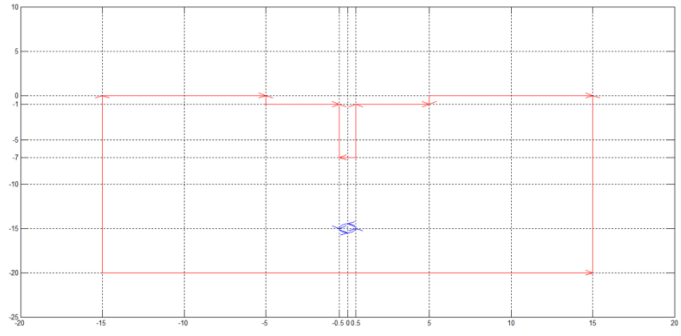


Figura 5.35.
Incorporación de un dren debajo de la presa

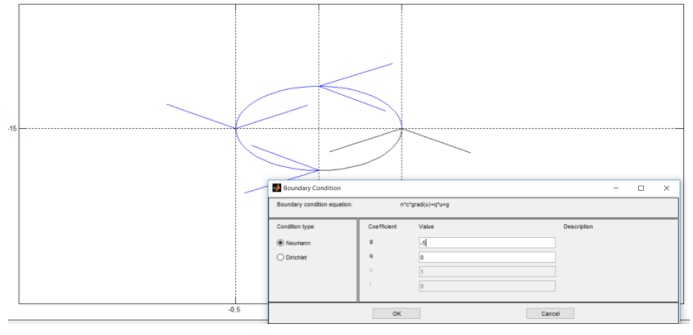


Figura 5.36.
Detalle de la Incorporación de un dren debajo de la presa

Asimismo, se pueden hacer diferentes simulaciones de caudales para analizar el comportamiento de las líneas de flujo, en este caso probamos con Q=20 y Q=40. Los resultados se muestran en las siguientes figuras.

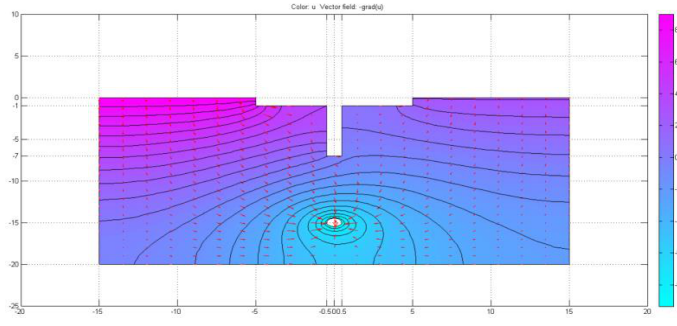


Figura 5.37.
Simulación con la incorporación de un dren debajo de la presa a Q=20

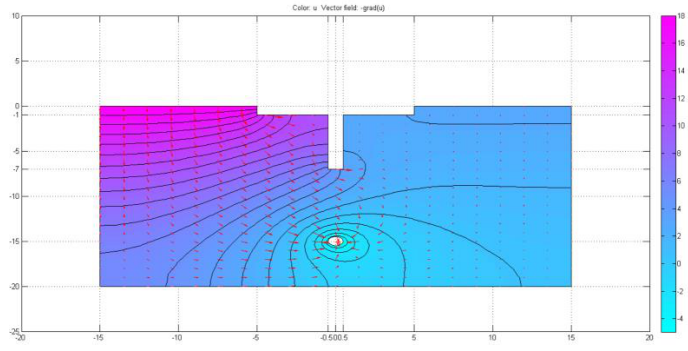


Figura 5.38.
Simulación con la incorporación de un dren debajo de la presa a Q=40

5.3.5. Perímetros de Protección de Pozos

Este caso consiste en determinar los perímetros (radio de influencia) de protección de pozos de explotación de agua subterránea, en régimen permanente, utilizando la herramienta PDETool de MATLAB.

Construir una grilla de 2000 m de largo por 1000 m de ancho. Utilizar una condición de carga o cabeza constante en el costado izquierdo ($H = 45$ m) y en el costado derecho del modelo ($H = 43$ m). Los costados superior e inferior tienen condición de borde de flujo nulo. Considere un tiempo máximo de un día, aunque la simulación es en régimen permanente. Asignar propiedades hidráulicas constantes en toda la malla. Asignar un pozo de bombeo con las coordenadas (1600, 500) y caudal de bombeo (-10 l/s).

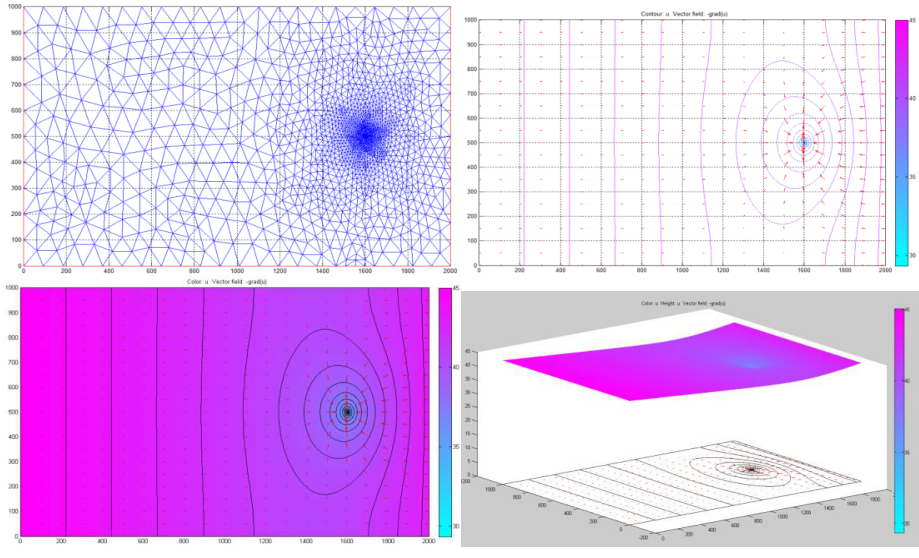


Figura 5.39.
Simulación de un pozo de protección en un sistema subterráneo

APÉNDICE A
INTRODUCCIÓN A MATLAB

Apéndice A
Introducción a MATLAB

Este apéndice de introducción a MATLAB ha sido elaborado tomando como base varios tutoriales de ayuda y la web MathWorks (https://la.mathworks.com/help/MATLAB/getting-started-with-MATLAB.html).

6.1. Introducción

Millones de ingenieros y científicos en todo el planeta utilizan MATLAB® para analizar y diseñar los sistemas y productos que transforman nuestro mundo. El lenguaje de MATLAB, basado en matrices, es la forma más natural del mundo para expresar las matemáticas computacionales. Las gráficas integradas facilitan la visualización de los datos y la obtención de información a partir de ellos. El entorno de escritorio invita a experimentar, explorar y descubrir. Todas estas herramientas y funciones de MATLAB están probadas rigurosamente y diseñadas para trabajar juntas.

MATLAB le ayuda a llevar sus ideas más allá del escritorio. Puede ejecutar sus análisis en conjuntos de datos de mayor tamaño y expandirse a *clusters* y nubes. El código de MATLAB se puede integrar con otros lenguajes, lo que le permite implementar algoritmos y aplicaciones en sistemas web.

6.2. Aspectos Fundamentales del Lenguaje

MATLAB es el acrónimo de "MATrix LABoratory" (laboratorio de matrices). Aunque otros lenguajes de programación, generalmente, procesan los números de uno en uno, MATLAB® funciona, principalmente, con matrices y arreglos completos. Los aspectos fundamentales del lenguaje incluyen operaciones básicas, como la creación de variables, la indexación de arreglos, operaciones aritméticas y tipos de datos. El objetivo de esta guía es explicar los comandos básicos para comenzar a programar en MATLAB.

6.3. Directorios

Antes de empezar, debemos indicarle al programa en qué directorio vamos a trabajar. Para esto vamos a File > Set Path y hacemos clic sobre el botón 'Add Folder'. Una vez que hayamos seleccionado la carpeta de trabajo, cerramos la ventana haciendo clic en 'Close'. En este caso hemos escogido trabajar en la unidad D, carpeta "ejemplos".

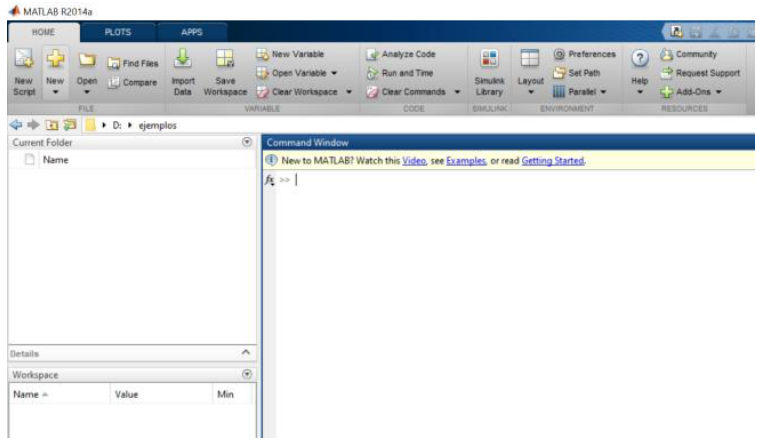


Figura A1.
Pantalla de inicio de MATLAB

6.4. Definición de Variables y Operatoria Básica

Para definir una variable en MATLAB, simplemente la declaramos de la siguiente forma: $a = 6$, a lo que el programa responderá desplegando el valor de "a" en pantalla de la siguiente manera:

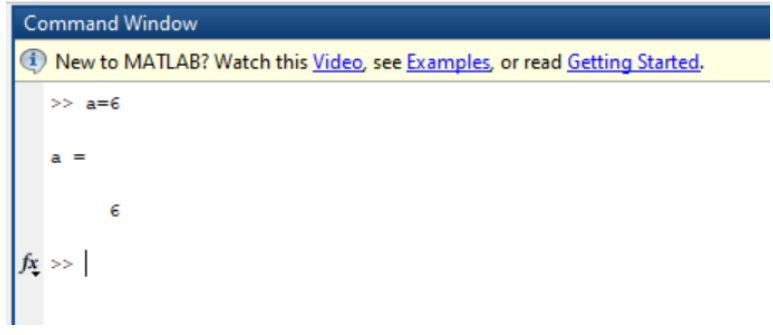


Figura A2.
Definición de variables y operaciones en MATLAB

Para evitar que en pantalla se despliegue un resultado, se agrega un punto y coma al final de la operación:

$a=6;$

Como pueden observar, no es necesario declarar el tipo de datos que contiene la variable a; a menos que se indique lo contrario, MATLAB trabaja por defecto con arreglos. En el ejemplo, a queda definido como una matriz de 1x1 cuyo único elemento es el número 6.

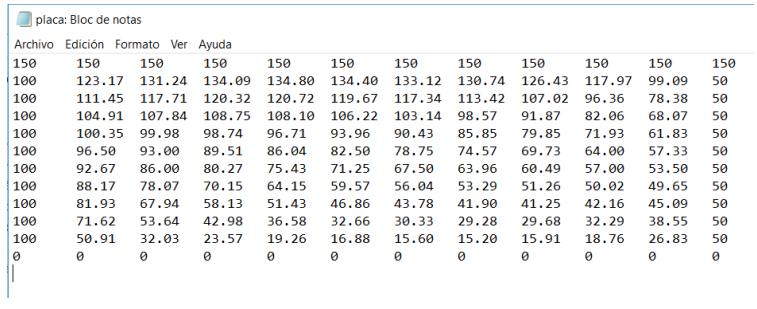
6.4.1. Declaración de Matrices

Los elementos de cada fila se separan por comas o espacios, para terminar con una fila y empezar la siguiente se pone un punto y coma. Ejemplos:

$A = [1,2,3;4,5,6;7,8,9]$ $\quad B = [1,2,3]$ $\quad C = [1;4;9]$

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \qquad C = \begin{bmatrix} 1 \\ 4 \\ 9 \end{bmatrix}$$

También se pueden crear vectores en forma rápida utilizando V=inicio: paso: fin
V = 1:0.2:2; genera V = [1 1.2 1.4 1.6 1.8 2]
o crear vectores equiespaciados utilizando "linspace" (inicio, fin, n.º puntos)
V = linspace (1,2,6); genera V = [1 1.2 1.4 1.6 1.8 2]
También es posible importar matrices desde Excel o archivo "txt"; por ejemplo, los siguientes datos los tenemos en el bloc de notas; archivo con nombre "placa.txt".


placa: Bloc de notas

Archivo Edición Formato Ver Ayuda

150	150	150	150	150	150	150	150	150	150	150	150
100	123.17	131.24	134.09	134.80	134.40	133.12	130.74	126.43	117.97	99.09	50
100	111.45	117.71	120.32	120.72	119.67	117.34	113.42	107.02	96.36	78.38	50
100	104.91	107.84	108.75	108.10	106.22	103.14	98.57	91.87	82.06	68.07	50
100	100.35	99.98	98.74	96.71	93.96	90.43	85.85	79.85	71.93	61.83	50
100	96.50	93.00	89.51	86.04	82.50	78.75	74.57	69.73	64.00	57.33	50
100	92.67	86.00	80.27	75.43	71.25	67.50	63.96	60.49	57.00	53.50	50
100	88.17	78.07	70.15	64.15	59.57	56.04	53.29	51.26	50.02	49.65	50
100	81.93	67.94	58.13	51.43	46.86	43.78	41.90	41.25	42.16	45.09	50
100	71.62	53.64	42.98	36.58	32.66	30.33	29.28	29.68	32.29	38.55	50
100	50.91	32.03	23.57	19.26	16.88	15.60	15.20	15.91	18.76	26.83	50
0	0	0	0	0	0	0	0	0	0	0	0

Figura A3.
Datos en bloc de notas

Luego podemos ejecutar el comando "load" y asignar esta matriz a la variable "A", la cual tendrá dimensiones 12 x 12:

A=load ('placa.txt'); % Se importa los datos de la matriz respuesta

6.4.2. Indexación de Elementos de una Matriz

Si definimos A como una matriz de m x n, podemos acceder a sus elementos a través de los siguientes comandos:

$A(i,j)$: Muestra el elemento de la fila i y columna j.
$A(i,:)$: Muestra la fila i.
$A(:,j)$: Muestra la columna j.

$A([1\ 3],j)$: Muestra los elementos de las filas 1 y 3 de la columna j.
$A(i,[1\ 4])$: Muestra los elementos de las columnas 1 y 4 de la columna i.

También se puede asignar valores a elementos específicos de una matriz, combinando los comandos de declaración e indexación; por ejemplo:

$A(1,2)$=17 : Guarda el número 17 en la fila 1, columna 2.
$A(i,:)$=17 : Guarda el número 17 en todos los elementos de la fila i.
$A(:,j)$=17 : Guarda el número 17 en todos los elementos de la columna j.

6.4.3. Matrices Especiales

En ciertas ocasiones se necesita definir matrices especiales, como identidades, matrices nulas, unitarias, diagonales, tridiagonales, etc. A continuación, ejemplos de estas matrices:

A': Matriz Transpuesta de A
eye(n) : Crea una matriz de identidad de n x n.
eye(m,n) : Crea una matriz de ceros de m x n con *unos* en su 'diagonal'.
diag(V): Crea una matriz diagonal con los elementos del vector V.
diag(V,n): Crea una matriz 'diagonal desplazada' en n con los elementos del vector V. Si n>0 la diagonal se desplaza hacia la derecha, si n<0 a la izquierda.
diag(A): Crea un vector con la diagonal de la matriz A.
ones(m,n): Crea una matriz de unos de m x n.
zeros(m,n): Crea una matriz de ceros de m x n.
rand(m,n): Crea una matriz de m x n de valores aleatorios.

Existen muchas más formas para definir matrices, y las que antes fueron mencionadas poseen más funciones. Para conocer más acerca de estas se recomienda utilizar la ayuda de MATLAB.

6.4.4. Operaciones con Matrices

Disponemos de las siguientes operaciones matriciales:

$A*B$	: Multiplicación matricial de A y B.
$A.*B$	: Multiplicación de los términos de A por los de B (conmutativa).
A/B	: Multiplicación matricial de A por la inversa de B.
$A\backslash B$	: Multiplicación matricial de B por la inversa de A.
$A./B$	: División de los términos de A por los de B.
$A+B$	: Suma de los términos de A con los de B.
$A-B$	: Resta de los términos de A con los de B.

Si U y V son vectores (matrices de una fila):

dot(U,V) : Producto punto de U con V ($\sum u_i . v_i$)
cross(U,V) : Producto cruz de U con V.

6.5. M-Files

En MATLAB se programa en M-Files, que son archivos de texto con una secuencia de instrucciones que luego se ejecutan en el programa. Para crear uno nuevo vamos a File > New > M-File.

6.5.1. Creación de Funciones

En MATLAB podemos crear funciones que ejecuten determinadas secuencias, las cuales pueden ser llamadas desde otro M-File. Para crear una nueva función debemos declararla como tal:

function [a,b] = nombre(c,d)

Donde a y b son las variables de salida (lo que entrega) y c y d son las variables de entrada (lo que recibe para poder ser ejecutada).
Ejemplo:

```
function [s,r] = sumaResta(a,b)
s=a+b;
r=a-b;
```

Guardamos el archivo en el directorio definido al comienzo y vamos a "Command Window" donde escribimos:

```
>> [a,b]=sumaResta(3,23)
a =
    26
b =
   -20
```

También es posible incluir comentarios en los M-Files. Para esto anteponemos % a lo que se quiera comentar. Si anotamos comentarios justo después del nombre de la función, esto se mostrará como ayuda, si escribimos el comando "help" nombre de la función en el "Command Window". Ejemplo:

```
function [s,r] = sumaResta(a,b)
%sumaResta(a,b) retorna un vector con la suma de a y b en su
%primer componente y la resta de ambos en el segundo.
s=a+b;      %Suma
r=a-b;      %Resta
```

En "Command Window":

```
>> help sumaResta
 sumaResta retorna un vector con la suma de a y b en su primer componente
y la resta de ambos en el segundo.
```

6.5.2. Operadores Lógicos

El operador "For" genera ciclos con incrementos de una variable; por ejemplo, crea el vector a = [2 4 6 ….20]:

```
for i=1:10
   a(i)=2*i;
end
```

La secuencia de instrucciones bajo el "if" se ejecutará solo si se cumple la condición especificada; por ejemplo:

```
for i=1:10
   a(i)=2*i;
   if a(i)>=10
      b(i)=-a(i);
   end
end
```

Crea los vectores a = [2 4 6 … 20] y b = [-10 -12 -14 … - 20]. También existen los condicionales "elseIf" y "else". A continuación, se muestra una lista con los operadores relacionales más utilizados:

Operador	Significado
>	Mayor que
<	Menor que
>=	Mayor o igual
<=	Menor o igual
&	Y (and)
|	O (or)

6.5.3. Gráficos

MATLAB permite graficar los resultados obtenidos. Para esto utilizamos el comando "plot". Además, existe una serie de funciones que aportan información adicional al gráfico: "plot" (vector x, vector y, opciones):

- Grafica los pares ordenados (vector $x\ (i)$, vector $y\ (i)$): Por defecto, MATLAB une los puntos que se grafican, pero esto se puede deshabilitar utilizando las opciones del comando "plot". Las opciones van desde cambiar los colores hasta el tipo de figura que se desea para marcar los puntos.
- xlabel('nombre'): Da el título al eje coordenado x.
- ylabel('nombre'): Da el título al eje coordenado y.
- title('nombre'): Da el título al gráfico.
- grid on: Hace visible la rejilla del gráfico.
- hold on: Mantiene el gráfico anterior mientras se hace uno nuevo
- figure: Crea una nueva ventana de gráfico
- legend('nombre'): Identifica el gráfico en pantalla (etiqueta el gráfico)

Ejemplos:
i) Graficaremos el movimiento oscilatorio amortiguado que tiene una estructura de un grado de libertad (Esto es una versión muy simplificada de cómo oscila un edificio) cuya ecuación es:

$$x(t) = \mathrm{A} \cdot \exp(-\beta \cdot \omega \cdot t) \cdot \mathrm{sen}(\omega \cdot t)$$

Donde
$A\,[\mathrm{m}]$: Amplitud
β: Coeficiente de amortiguamiento
$\omega\,[1/\mathrm{s}]$: Frecuencia natural de oscilación de la estructura
$t\,[s]$: Tiempo

A continuación, presentamos un M-File que resuelve el problema:

```
%Definimos el vector de tiempo:
t=0:1/200:10;

%Definimos los parámetros de la ecuación:
w=2*pi;     %Frecuencia
b=0.05;     %Amortiguamiento
A=3;        %Amplitud

%Calculamos la ecuación de movimiento:
y=A*exp(-b*w*t).*sin(w*t);      %va con .* ya que la exponencial y el
                                %seno son vectores que evalúan la serie
                                %de tiempo

%Graficamos:
plot(t,y)
title('Decaimiento Libre')
xlabel('tiempo [s]')
ylabel('Amplitud [cm]')
legend('Ampliad')
grid on
```

Guardamos y corremos el programa, resultando el siguiente gráfico:

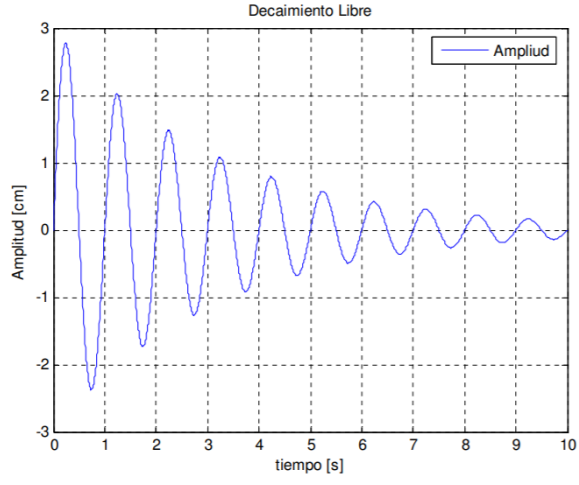


Figura A4.
Ejemplo de salida gráfica en MATLAB con una función

ii) Grafiquemos ahora dos movimientos con distinta amplitud en un mismo gráfico:

```
%Definimos el vector de tiempo:
t=0:1/200:10;

%Definimos los parámetros de la ecuación:
w=2*pi;    %Frecuencia
b=0.05;    %Amortiguamiento
A=3;       %Amplitud uno
B=1.5;      %Amplitud dos

%Calculamos la ecuación de movimiento:
y1=A*exp(-b*w*t).*sin(w*t);    %va con .* ya que la exponencial y el
                               %seno son vectores que evalúan la serie
                               %de tiempo

%Calculamos la segunda ecuación:
y2=zeros(length(t));       %Damos dimensiones al vector creando un
                           %vector de ceros. Esto hace mucho más rápido
                           %el programa.

%Calculamos la ecuación de movimiento:
for i=1:length(t)
   y2(i)=B*exp(-b*w*t(i))*sin(w*t(i));
end

%Graficamos:
plot(t,y1)
title('Decaimiento Libre')
xlabel('tiempo [s]')
ylabel('Amplitud [cm]')
grid on
hold on
plot(t,y2,'Red')
legend('Amplitud=3','Amplitud=1.5');
```

El gráfico resultante es:

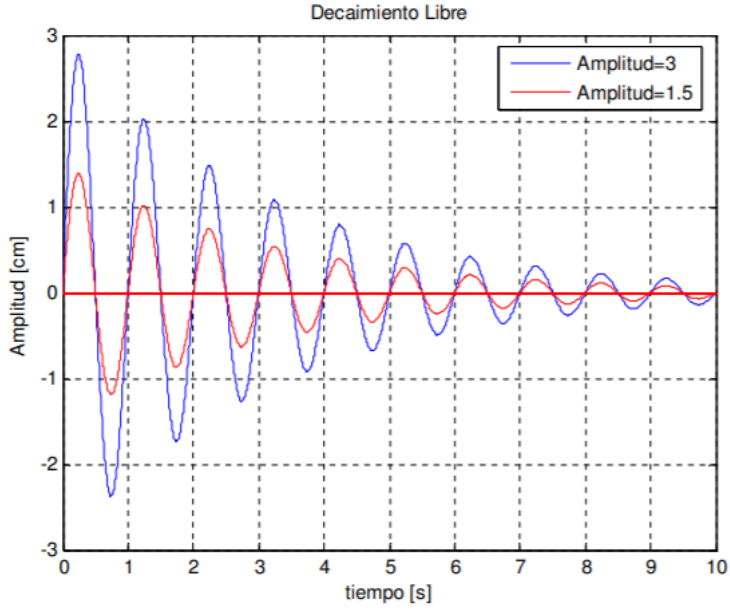


Figura A5.
Ejemplo de salida gráfica en MATLAB con dos funciones

Probemos ahora la opción 'subplot' para graficar. Si reemplazamos el código que grafica por:

```
%Graficamos:
subplot(2,1,1)
plot(t,y1)
title('Decaimiento Libre')
xlabel('tiempo [s]')
ylabel('Amplitud [cm]')
legend('Amplitud=3')
grid on
subplot(2,1,2)
plot(t,y2,'Red')
title('Decaimiento Libre')
xlabel('tiempo [s]')
ylabel('Amplitud [cm]')
legend('Amplitud=1.5')
grid on
```

La salida gráfica será:

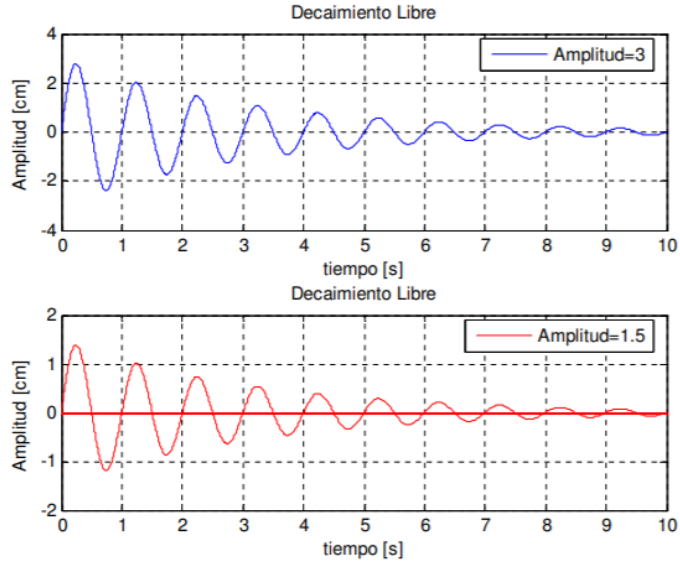


Figura A6.
Ejemplo de salida gráfica tipo 'subplot' en MATLAB

6.6. Acerca de la Ayuda de MATLAB

MATLAB cuenta con una excelente plataforma de ayuda al usuario. Es fundamental utilizarla, ya que es imposible poner en una guía todo lo que se puede hacer con este programa. Alguien que utiliza bien MATLAB no es aquel que se sabe todas las funciones de memoria, sino quien sabe usar de forma eficiente la ayuda.

Para llamar a la ayuda vamos a Help > MATLAB Help o presionamos F1. Les recomiendo que utilicen la pestaña "index" o "search" para buscar temas en la ayuda. Si quieren ver detalles de una función, pongan ahí el nombre de la función. Si quieren hacer algo, pero no saben si existe la función que lo hace, busquen por las "palabras clave" con términos relacionados a lo que quieren hacer. Es muy probable que encuentren lo que están buscando.

A continuación, se lista las funciones que les pueden ser útiles en su vida estudiantil, busquen para qué sirven en la ayuda.

cla
clc
cumsum
cumtrapz
disp

eig
fft
getframe
guide
input
length
max
min
movie2avi
nargin
nargout
plot
plot3
roots
size
sort
sparse
sum

APÉNDICE B
INTRODUCCIÓN A GUIDE DE MATLAB

7.1. Introducción

Una interfaz gráfica es el vínculo entre el usuario y un programa computacional, constituida, generalmente, por un conjunto de comandos o menús, instrumentos y métodos por medio de los cuales el usuario se comunica con el programa durante las operaciones que se desean realizar, facilitando la entrada y salida de datos e información. Una interfaz es una de las partes más importantes de cualquier programa, puesto que determina qué tan factible y preciso será el desempeño del programa ante los comandos que el usuario pretenda ejecutar. Aunque un programa sea muy poderoso, si se manipula por medio de una interfaz pobremente elaborada, tendrá poco valor para un usuario inexperto. Es por esto que las interfaces gráficas tienen una gran importancia para usuarios inexpertos o avanzados de cualquier programa ya que facilitan su uso.

Ejemplos de interfaces gráficas son las ventanas de Word, Excel, la ventana de MATLAB, entre otras. Una interfaz gráfica consta de botones, menús, ventanas, etc., que permiten utilizar de una manera muy simple, y en ocasiones casi intuitiva, programas realizados en ambientes como Windows y Linux. Las interfaces gráficas también se conocen como interfaces de usuario. El nombre en inglés de las interfaces gráficas es Graphical User Interface y se denominan GUI, por lo que nosotros también nos referiremos a ellas de la misma manera.

Existen diferentes lenguajes de programación que permiten crear GUI, tales como Visual C, Visual Basic, TK y MATLAB, por mencionar algunos. Todos ellos permiten usar diferentes controles y tienen distintas maneras de programarlos. MATLAB nos permite realizar GUI de una manera sencilla, usando una herramienta llamada GUIDE (GUI Development Environment). En este apéndice presentaremos una introducción muy completa a las técnicas en MATLAB para crear interfaces gráficas.

GUIDE (Graphical User Interface Development Environment) es un juego de herramientas que se extiende por completo en el soporte de MATLAB, diseñadas para crear GUI (Graphical User Interfaces) fácil y rápidamente, prestando ayuda en el diseño y presentación de los controles de la interfaz, reduciendo la labor al grado de seleccionar, tirar, arrastrar y personalizar propiedades. Una vez que los controles están en posición se editan las funciones de llamada ("Callback") de cada uno de ellos, escribiendo el código de MATLAB que se ejecutará cuando el control sea utilizado. Siempre será difícil diseñar GUI, pero no debería ser difícil implementarlas.

7.2. Diseño del GUIDE

GUIDE está diseñado para hacer menos tedioso el proceso de aplicación de la interfaz gráfica y obviamente para trabajar como herramienta de trazado de GUI. Entre sus poderosos componentes está el editor de propiedades ("property editor"), este se encuentra disponible en cualquier momento que se esté lidiando con los controles

de MATLAB. El editor de propiedades por separado se puede concebir como una herramienta de trazado y asistente de codificación (revisión de nombres y valores de propiedades). Cuando se fusiona con el panel de control, el editor de menú, y la herramienta de alineación, resulta una combinación que brinda un inigualable control de los gráficos en MATLAB.

El beneficio que proporciona el uso de GUI es evidente, ya que permite al usuario ejecutar, cómodamente, el código desarrollado en MATLAB, sin necesidad de cumplir la incómoda sintaxis funcional necesaria cuando se trabaja desde la línea de órdenes. A diferencia de la ejecución de funciones o "scripts" de MATLAB, la ejecución de GUI no predetermina el flujo de ejecución del código. Es el usuario, a través de su interacción con el GUI, el que determina el orden en que se ejecutan las diferentes órdenes y funciones desarrolladas. Otra diferencia importante es que la ejecución no termina cuando finaliza la ejecución del "script" o función, sino que el GUI permanece abierto, permitiendo al usuario invocar la ejecución de ese u otro código desarrollado.

El desarrollo de GUI se realiza en dos etapas:
- Diseño de los componentes (controles, menús y ejes) que formarán el GUI.
- Codificación de la respuesta de cada uno de los componentes ante la interacción del usuario.

7.3. Iniciando GUIDE

A la herramienta GUIDE se accede de varias maneras, la primera de ellas es tecleando guide en la ventana de comando.

>> guide

Otra manera de entrar en GUIDE, es a través de la opción "File", haciendo clic en "New" y por último eligiendo la opción GUI, (como se muestra en la figura).

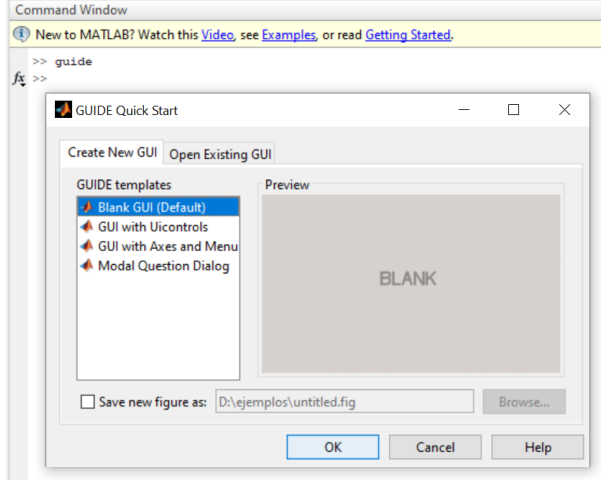


Figura B1.
Pantalla de inicio GUIDE de MATLAB

Esta ventana de diálogo, correspondiente con la ventana de inicio de GUI, presenta las siguientes opciones:

a) *Blank GUI (Default):* La opción de interfaz, gráfica de usuario en blanco (viene predeterminada), nos presenta un formulario nuevo, en el cual podemos diseñar nuestro programa.

b) *GUI with Uicontrols:* Estos controles de la interfaz de usuario son componentes, como botones y controles deslizantes, con los que los usuarios pueden interactuar. La "uicontrolfuncion" crea un control de interfaz de usuario y establece las propiedades requeridas antes de mostrarlo. Al cambiar los valores de propiedad puede modificar la apariencia y el comportamiento de los controles de la interfaz de usuario. Use la notación de puntos para referirse a un objeto y propiedad específicos.

c) *GUI with Axes and Menu:* Esta opción contiene el menú "File" con las opciones "Open", "Print" y "Close". En el formulario tiene un "Popup menu", un "push button" y un objeto "Axes". Podemos ejecutar el programa eligiendo alguna de las seis opciones que se encuentran en el menú despegable y haciendo clic en el botón de comando.

d) *Modal Question Dialog:* Con esta opción se muestra en la pantalla un cuadro de diálogo común, el cual consta de una pequeña imagen, una etiqueta y dos botones ("Yes" y "No"), dependiendo del botón que se presione el GUI retorna el texto seleccionado (la cadena de caracteres "Yes" o "No"). Si elegimos la primera opción, Blank GUI, tenemos:

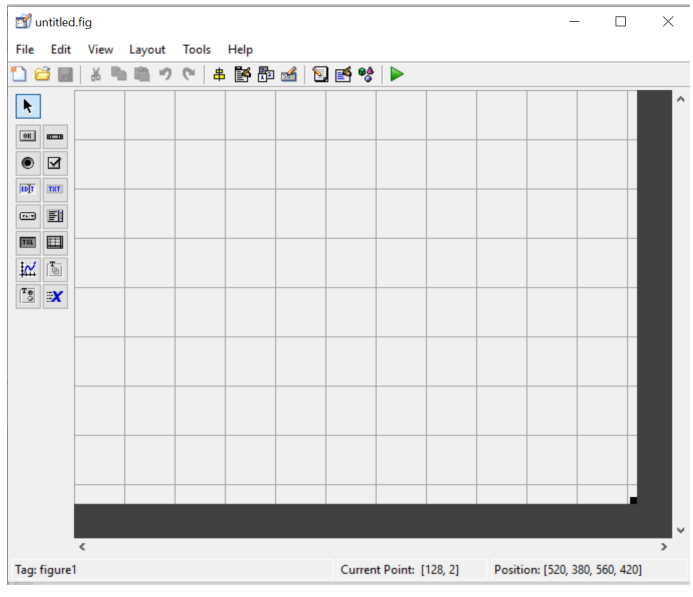


Figura B2.
Formulario GUIDE de MATLAB

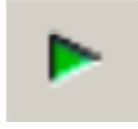
Los componentes principales de GUIDE son:

Barra de Menús: Aquí se encuentran las funciones elementales de Edición de GUI.

- Paleta de Componentes ("component Palette"): Aquí se encuentran los "uicontrols", estos componentes permiten seleccionar los controles (objetos) que son los que se muestran en la figura.

- La Barra de Herramientas: En ella se encuentran los siguientes botones:

  (1) Botón de ejecución ("Run button"): Al presionarse crea la figura de la interfaz diseñada en el "Layout Area".

  (2) Alineación de componentes ("Alignment tool"): Esta opción permite alinear los componentes que se encuentran en el área de trabajo ("Layout Area") de manera personalizada.

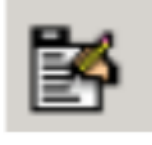
  (3) Inspector de propiedades ("Property Inspector"): Con esta opción se asignan y modifican las propiedades de cada objeto en forma personalizada.

  (4) Navegador de objetos ("Object Browser"): Muestra todos los objetos que se encuentran en la figura (en forma de árbol) y a través del "Object Browser" se pueden seleccionar los objetos.

  (5) Editor de menús ("Menu Editor"): El redactor de menú crea menús de ventana y menús de contexto.

7.4. Flujo de Operación con GUI

Con una GUI, el flujo de cómputo está controlado por las acciones en la interfaz. Mientras que en un "script" el flujo de comandos está predeterminado, el flujo de operaciones con una GUI no lo está. Los comandos para crear una interfaz con el usuario se escriben en un "script", la interfaz invoca que se ejecute el "script", mientras la interfaz del usuario permanece en la pantalla, aunque no se haya completado la ejecución del "script".

En la siguiente figura se muestra el concepto básico de la operación del software con una GUI. Cuando se interactúa con un control, el programa registra el valor de esa opción y ejecuta los comandos prescritos en la cadena de invocación. Los menús de interfaz con el usuario, los botones, los menús desplegables, los controladores deslizantes y el texto editable son dispositivos que controlan las operaciones del software. Al completarse la ejecución de las instrucciones de la cadena de invocación, el control vuelve a la interfaz para que pueda elegirse otra opción del menú. Este ciclo se repite hasta que se cierra el GUI.

El control guarda un "string" que describe la acción a realizar, cuando se invoca puede consistir en un solo comando de MATLAB o una secuencia de comandos, o en una llamada a una función. Es recomendable utilizar llamadas a funciones, sobre todo cuando se requiere de más de unos cuantos comandos en la invocación.

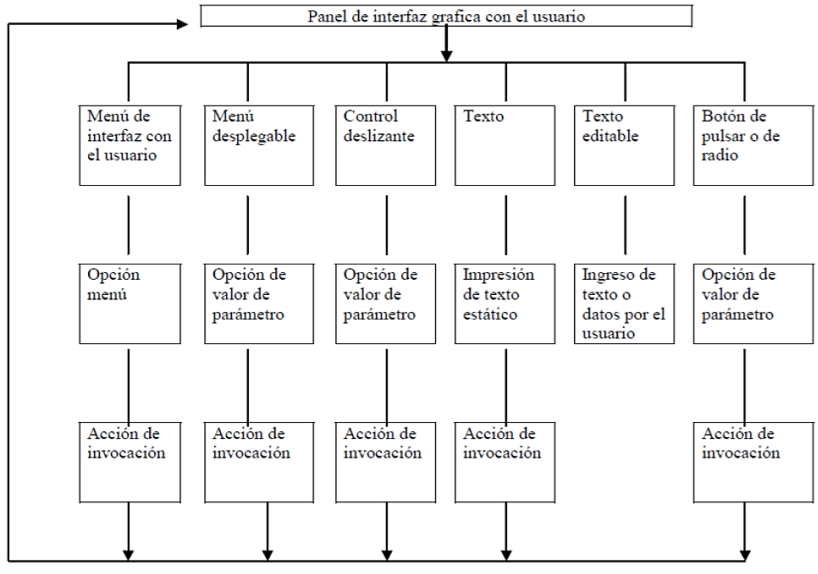


Figura B3.
Esquema de la interfaz gráfica de GUIDE de MATLAB

Básicamente solo se necesitan entender cinco comandos para poder describir una GUI: "uimenu", "uicontrol", "get", "set" y "axes". No obstante, lo que hace relativamente complicado a estos comandos es el gran número de formas de uso que tienen. Es imposible describir todos los tipos de situaciones, pues requiere demasiado espacio y sería muy laborioso leerlo. Por tanto, solo trataremos de explicar los elementos básicos de una GUI.

7.5. Organización de los Objetos Gráficos

El programa MATLAB ofrece una serie de funciones de dibujo y visualización de datos, como es el caso de "imshow", "imagesc" o "plot", por ejemplo. Cuando llamamos a una función gráfica, MATLAB crea el gráfico requerido usando objetos como ventanas, ejes de coordenadas, líneas, texto, etc. Podemos trabajar con gráficos en MATLAB a tres niveles diferentes:

- Realizando llamadas a funciones de dibujo y visualización. MATLAB muestra todas las gráficas en un tipo de ventanas especiales conocidas como "figures", en las que se sitúan unos ejes de coordenadas. Estos ejes son los que proporcionan el sistema de coordenadas necesario para realizar la visualización de los datos.

- Trabajando a "bajo nivel", haciendo las llamadas necesarias a funciones de MATLAB para ir creando los objetos gráficos que sean necesarios para hacer la visualización. A este nivel se realizarán múltiples llamadas a la función "set" y otras similares para establecer las propiedades de los distintos objetos gráficos que se vayan creando o para modificarlas durante la ejecución del programa. Este modo de trabajar es muy similar al uso tradicional de una biblioteca gráfica que se puede hacer en un lenguaje de programación convencional.

- Empleando el entorno de desarrollo de GUI, que nos permitirá definir todos los componentes gráficos que deseemos y establecer sus propiedades e incorporar código de respuesta a cada una de las acciones del usuario a través de una herramienta gráfica de cómodo manejo. Además, tiene la ventaja de que en cualquier momento podremos elegir si deseamos trabajar a alto o bajo nivel, pudiendo acceder al código asociado a través del editor de MATLAB.

Como podemos ver, en cualquiera de estos tres casos estamos haciendo uso del sistema de objetos gráficos de MATLAB. Estos objetos gráficos son los elementos básicos empleados por MATLAB para visualizar datos, y están organizados en una jerarquía como la de la siguiente figura, en la que se muestran los tipos de objetos gráficos empleados con más frecuencia:

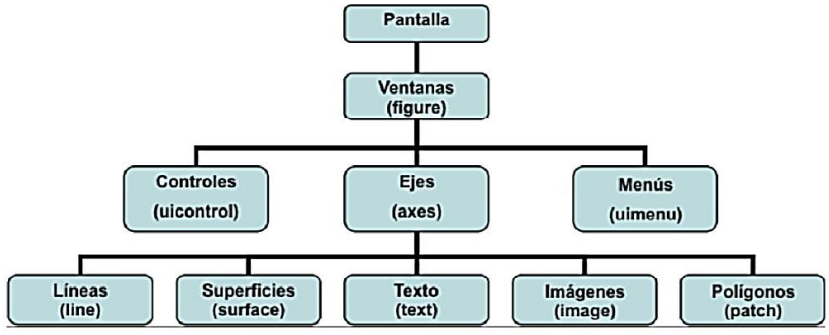


Figura B4.
Diagrama de organización de objetos gráficos en GUIDE de MATLAB

Como se ve en la figura anterior, el objeto más general es la pantalla. Es la raíz de la jerarquía. Una pantalla puede contener una o varias ventanas ("figure"). A su vez, cada una de las ventanas podrá contener controles ("uicontrol"), como son los botones, menús desplegables, etc., menús ("uimenu") y uno o más ejes de coordenadas ("axes") en los que se podrán representar objetos de nivel inferior. Los ejes pueden incluir cinco tipos de elementos gráficos: líneas ("line"), polígonos ("patch"), texto ("text"), superficies ("surface") e imágenes de mapa de "bits" ("image").

Al establecerse relaciones padre-hijo entre los objetos gráficos se facilita su gestión, ya que, por ejemplo, cuando se borra un objeto, se borrarán automáticamente todos los descendientes de este.

Cada objeto está asociado a un identificador ("handle") único desde el momento de su creación. A través de este identificador podremos modificar las características (llamadas propiedades del objeto) de un objeto gráfico. Naturalmente, también podremos establecer las propiedades de un objeto en el momento de su creación (cambiarlas con respecto a los valores por omisión). En el identificador del objeto raíz, la pantalla es siempre cero. El identificador de las distintas ventanas ("figure") es un entero que aparecerá en la barra de título de la ventana. Los identificadores de los demás objetos gráficos son números reales. Cualquier identificador se puede obtener como valor de retorno de una función y almacenarse en una variable.

Como hemos dicho, puede haber varias ventanas abiertas, pero solo una de ellas es la ventana activa en cada momento. De la misma forma, una ventana puede contener varios ejes de coordenadas, pero solo unos son los ejes activos. El objeto activo será el último objeto creado o sobre el que se haya hecho clic con el ratón. Podemos obtener los identificadores de la ventana, los ejes y el objeto activos con las órdenes:

- gcf: Devuelve un entero, el identificador de la ventana activa.
- gca: Devuelve el identificador de los ejes activos.
- gco: Devuelve el identificador del objeto activo.

La principal utilidad que tiene conocer los identificadores de los objetos gráficos es que a través de ellos podremos modificar las propiedades de los objetos o incluso borrarlos:

- set (id): Muestra en pantalla todas las propiedades del objeto al que corresponde el identificador id.
- get (id): Produce un listado de las propiedades y de sus valores.
- set (id, "propiedad", "valor"): Establece un nuevo valor para la propiedad del objeto con identificador "id". Se pueden establecer varias propiedades en la misma llamada a "set" incluyendo una lista de parejas "propiedad", "valor" en la llamada.
- get (id, "propiedad"): Obtiene el valor de la propiedad especificada.
- delete (id): Borra el objeto cuyo identificador es id y todos sus hijos.

7.6. Partes de GUIDE

A continuación, mencionaremos las partes más importantes de GUIDE y posteriormente explicaremos con más detalle. A continuación, mostramos una lista de los controles más habituales.

Listbox	Caja de lista	Toggle Button	Botón de activación
Frame	Marco	Radio Button	Botón de selección
Edit Text	Cuadro de edición	Checkbox	Botón de elección
Slider	Barra de deslizamiento	Push Button	Botón de presión
Static Text	Cuadro de texto	Popup Menu	Botón de aparición

Cada botón cuenta con una serie de propiedades que explicaremos en el apartado siguiente.

7.7. Propiedades de los Controles

Los controles de la interfaz con el usuario en MATLAB se especifican con la orden "uicontrol". Estos controles tienen mucho en común con los menús de la interfaz con el usuario, pero los primeros tienen muchos estilos. La sintaxis de "uicontrol" es:

```
k = uicontrol ('Style', 'especificación de estilo', ...
               'String', 'cadena para exhibir', ...
               'Value', [valor], ...
               'BackgroundColor', [r,g,b], ...
               'Max' [valor], ...
               'Min' [valor], ...
               'Position', [izq, base, ancho, alto], ...
               'CallBack', 'cadena de invocacion')
```

Donde "especificación de estilo" es una de las siguientes cadenas:

popup
push
radio
checkbox
slider
edit (texto editable)
text (texto estático)
frame

Las propiedades de iucontrol son similares a las de "uimenu". Destacamos:

a) "Value", valor: Especifica el valor por omisión de ajuste. En el caso de interruptores de encendido/apagado el valor es 0 o 1. En el caso de un control deslizante ("slider") puede ser cualquier valor entre el mínimo y el máximo.
b) "Min", Valor: Establece el valor mínimo. Su significado difiere dependiendo del estilo.
c) "Max", Valor: Establece el valor máximo. Su significado difiere dependiendo del estilo.

Hay muchas más propiedades que pueden incluirse en los comandos del "uicontrol", tal como sucede con las propiedades del "uimenu", aunque al programar conviene minimizar el número de propiedades a fin de simplificar el "sript".

Texto estático.

Un "static text" puede exhibir símbolos, mensajes o incluso valores numéricos de una GUI y puede colocarse en un lugar deseado. El texto estático no tiene cadenas de invocación. A continuación, mostramos un ejemplo de texto estático.

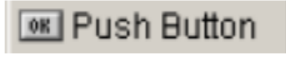
```
k1 = uicontrol (´Style´, ´text´, …
´String´, ´cadena para exhibir´, …
´Position´, [20, 50, 140, 30])
```

El contenido de un texto exhibido puede modificarse si es necesario. Esto se hace con el comando "set"; por ejemplo, si se ejecuta el comando que sigue desde la ventana de comandos mientras está vigente el ejemplo anterior de orden "uicontrol":

```
set (k1, ´string´, ´Ahora aparece un texto modificado´)
```

Menú desplegable.

Los "pop-up menus" difieren de los menús de interfaz con el usuario en que pueden aparecer en cualquier punto de la ventana de la figura, mientras que los menús de interfaz con el usuario solo se localizan en la parte superior.



Push button

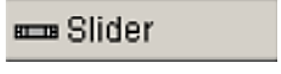
Los "Push button" generan una acción cuando hacemos clic con el puntero del ratón sobre ellos. Cuando se da clic en un "push button" aparece presionado; cuando se suelta el botón del *mouse*, el botón aparece levantado; y su rutina de llamada se ejecuta.

Casilla de verificación.

Las casillas de verificación están diseñadas para realizar operaciones de encendido/apagado. Las posiciones de encendido/apagado se registran en "Value" que puede examinarse con "get" ("handle", "value"). Los comandos "axis on" y "axis off" se escriben en la cadena de invocación.

Botón de radio.

Cuando solo se usa un botón de radio, no existe diferencia funcional alguna con respecto a una casilla de verificación. Por otro lado, los botones de radio en grupo son mutuamente exclusivos; es decir, si un botón está encendido, todos los demás botones se apagan, mientras que las casillas de verificación son independientes entre sí. Sin embargo, esta característica exclusiva de los botones de radio solo puede implementarse mediante la programación del usuario en la cadena de invocación.



Barra deslizadora.

Los "sliders" aceptan datos de entrada numéricos con un rango específico. Los usuarios mueven la barra dejando presionado el botón del mouse y arrastrándola, haciendo clic en la flecha. La posición de la barra indica un valor numérico.

Texto editable.

El dispositivo de texto editable permite al usuario teclear una cadena de entrada. Se pueden escribir varios valores numéricos en forma de vector o matriz como cadena mediante el mismo dispositivo; esta cadena se convertirá posteriormente en valores numéricos con el comando "str2num".

Un ejemplo de "uicontrol" para texto editable es:
ed1 = uicontrol (gcf, ´Style´, ´edit´, …
´Position´, [10, 260, 110, 20], …
´CallBack´, inp_txt = get (ed1, ´´string´´)´)

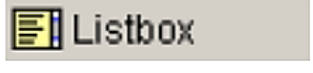
Las palabras clave en el comando anterior son: "Style", "edit" y "get", que capturan el texto introducido.

Marcos.

El estilo marcos puede servir para agrupar dispositivos como lo botones de radio o las casillas de verificación.

Botón de palanca.

El "toggle button" genera una acción que indica un estado binario (*on* u *off*). Cuando se hace clic en un "toggle button" aparece presionado y permanece así hasta que se suelta el botón del *mouse,* y en ese momento ejecuta la llamada. Un clic posterior del *mouse* regresa al "toggle button" a su estado original y vuelve a ejecutar la rutina de llamada.



Cajas de lista.

El componente "Listbox" muestra una lista de artículos, y permite a usuarios seleccionar uno o más artículos.

Múltiples ejes: Al crear una interfaz con el usuario, a menudo hay necesidad de trazar una o más gráficas dentro de la interfaz. Podemos usar el comando "subplot" para este fin, pero el comando "axes" es más flexible y ofrece opciones versátiles a los programadores. El comando "axes" abre un eje en un punto especificado dentro de una ventana de figura.

"Property inspector": El inspector de propiedades está compuesto de las siguientes propiedades o atributos, tal y como se muestra en la figura.

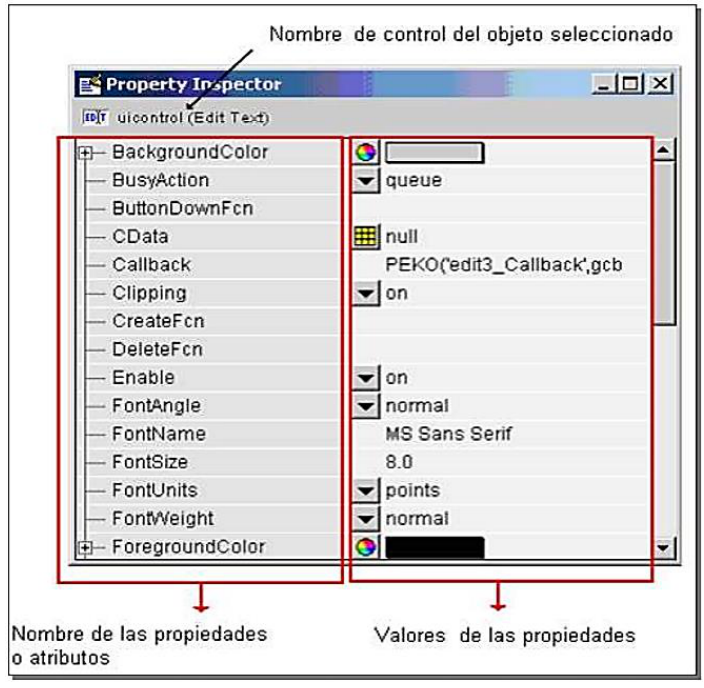


Figura B5.
Ventana emergente del inspector de propiedades de GUIDE de MATLAB

Cada uno de los controles tiene propiedades particulares, los cuales se analizan a continuación:

"BackgroundColor"
El color usado para rellenar el rectángulo de "unicontrol". Especifica un color usando un vector de tres elementos RGB (rojo, verde y azul) o uno de los nombres ya predefinidos en MATLAB. El color por "defecto" es determinado por la configuración del sistema.

"BusyAction"
Interrupción de la rutina de llamada ("callback"). Si una llamada es ejecutada y el usuario activa un evento en un objeto para el cual una llamada está definida, esa llamada trata de interrumpir la primera llamada. La primera llamada puede ser interrumpida solamente por uno de los siguientes comandos: "drawnow", "figure", "getframe", "pause" o "waitfor"; si la llamada no contiene ninguno de estos comandos no puede ser interrumpida.

Si la propiedad interrumpible del objeto que está ejecutando la llamada está desactivada (*off*), la llamada no puede ser interrumpida (excepto por algunas llamadas). La propiedad "BusyAction" del objeto cuya llamada está esperando para ejecutarse determina lo que le pasa a la llamada:

- Si el valor es "queue", la llamada es agregada al evento "queue" y se ejecuta después de que la primera llamada termina de ejecutarse.
- Si el valor es "Cancel", el evento es descartado y la llamada no se ejecuta.

Si la llamada interrumpida es una llamada de "DeleteFcn" o "CreateFcn" o una de una figura de "CloseRequest" o "ResizeFcn" se interrumpe y ejecuta sin importar el valor de la propiedad interrumpible del objeto.

"ButtonDownFcn"
Una rutina de llamada que se ejecuta cuando se presiona un botón del *mouse* mientras el cursor está en un "uicontrol". Cuando la propiedad "enable" del "unicontrol" está desactivada, el "ButtonDownFcn" se ejecuta al hacer clic en el "unicontrol". Esto es útil para implementar acciones para modificar interactivamente las propiedades de control del objeto, como el tamaño y la posición.

Esta rutina se define como una cadena ("string") que es una expresión válida en MATLAB o el nombre de un archivo M (M-file). La expresión se ejecuta en el espacio de trabajo de MATLAB. La propiedad de llamada define la rutina de llamada que se ejecuta cuando se hace clic en el botón.

"Callback"
Controla la acción. Una rutina que se ejecuta cuando se activa un objeto de la clase "uicontrol". Define esta rutina como una cadena. La expresión se ejecuta en el espacio de trabajo de MATLAB. Para ejecutar la rutina para un control de texto editable, escribe el texto deseado y después sigue uno de los siguientes pasos:

- Mueve la selección del objeto (da clic en cualquier otra parte).
- Para un texto editable de una sola línea presiona "return".
- Para una caja de texto ("text box") presiona "Ctrl-Return".

Esta rutina definida para los componentes "frame" y "ststic text" no se ejecuta, porque ninguna acción está asociada con estos objetos.

"Cdata"
Imagen de color verdadero mostrada en un control. Una matriz tridimensional de valores RGB que definen una imagen de color verdadero que es mostrada ya sea en un "push button" o un "toggle button". Cada valor debe tener un rango entre cero y uno.

"CreateFcn"
Rutina de llamada ejecutada cuando se crea un objeto. Esta propiedad define una rutina de llamada que es ejecutada cuando MATLAB crea un objeto de la clase "uicontrol". Se debe definir esta propiedad como un valor por defecto para los "uicontrols".

"DeleteFcn"
Una rutina de llamada que se ejecuta cuando se borra un objeto "uicontrol". MATLAB ejecuta la rutina antes de destruir las propiedades del objeto, así sus valores están disponibles para la rutina de llamada.

"Enable"
Activa o desactiva el "uicontrol". Esta propiedad controla cómo los "uicontrols" responden a un clic del *mouse,* incluyendo qué rutina de llamada se ejecuta.

- *on* - El "uicontrol" es operacional.
- *inactive* - no es operacional pero se ve como si estuviera activado.
- *off* - No es operacional y su etiqueta se vuelve gris.

Cuando se hace clic con el botón izquierdo del *mouse* sobre un objeto "uicontrol" que tiene su propiedad "enable" activada (en *on*), MATLAB ejecuta las siguientes acciones:

- Asigna la propiedad de la figura "SelectionType".
- Ejecuta la rutina de llamada del control.
- No asigna la propiedad de la figura "CurrentPoint" y tampoco ejecuta ni la propiedad de control "ButtonDownFcn" ni la rutina de llamada de la figura "WindowButtonDownFcn".

Cuando se hace clic con el botón izquierdo del *mouse* sobre un objeto "uicontrol" que tiene su propiedad "enable" inactiva, o cuando se hace clic derecho en uno en el que "enable" tiene cualquier valor, MATLAB ejecuta estas acciones en este orden:

(1) Asigna la propiedad de la figura "SelectionType".
(2) Asigna la propiedad de la figura "CurrentPoint".
(3) Ejecuta la rutina de llamada "WindowButtonDownFcn" de la figura.
(4) En un clic derecho, si el "uicontrol" está asociado con un "context menú", registra el "context menu".
(5) Ejecuta la llamada "ButtonDownFcn" del control.
(6) Ejecuta la llamada del elemento seleccionado del "context menu".
(7) No ejecuta la rutina de llamada del control.
(8) Poniendo esta propiedad inactiva te capacita para implementar arrastre o cambio de tamaño de objetos usando la rutina de llamada.
(9) "ButtonDownFcn".

"Extent"
Tamaño de un carácter de cadena "uicontrol". Un vector de cuatro elementos que define el tamaño y la posición de un carácter de tipo cadena usado para etiquetar el "uicontrol" tiene la forma:

[0, 0, width, height]

Los dos primeros elementos siempre son cero. "width" (ancho) y "height" (alto) son las dimensiones del rectángulo. Todas las medidas son unidades especificadas por la propiedad "Units".

Ya que la propiedad "Extent" está definida en las mismas unidades que el "uicontrol", se puede usar esta propiedad para determinar el tamaño adecuado del ancho del "uicontrol", con respecto a su etiqueta, haciendo lo siguiente:

- Definiendo la propiedad "String" y seleccionando la fuente usando las propiedades relevantes.
- Tomando el valor de la propiedad "Extend".
- Definiendo "width" y "height" de la propiedad "Position" (posición) se logra mayor amplitud que "width" y "height" de "Extend".

"FontAngle"
Inclinación de un carácter. Poniendo esta propiedad en *Italic* (itálica) u oblique (oblicua) selecciona una versión inclinada de la fuente, cuando está disponible en el sistema.

"FontName"
El nombre de la fuente que mostrará la cadena para visualizar e imprimir correctamente debe ser un tipo de fuente que el sistema soporte. Para usar un ancho ajustado que se vea bien en cualquier exterior (y que se muestre correctamente en Japón, donde usan caracteres "multibyte") se debe poner al "FontName" la cadena: "FixedWidth" (esta cadena es sensible a las mayúsculas): "Set" ("uicontrol_handle", "FontName", "FixedWidth").

"FontSize"
Tamaño de la fuente. Un número que especifica el tamaño de la fuente que va a ser mostrado en la cadena en unidades determinadas por la propiedad "FontUnits". El tamaño por defecto es dependiente del sistema.

"FontUnits"
Unidades del tamaño de la fuente. Esta propiedad determina las unidades usadas por la propiedad "FontSize". Las unidades normalizadas interpretan el "FontSize"

como una fracción de la altura del "uicontrol". Cuando se cambia el tamaño del "uicontrol", MATLAB modifica la pantalla "FontSize", "pixels" (pixeles), "inches" (pulgadas), "centimeters" (centímetros) y "points" (puntos) son unidades absolutas (1 punto = 1/72 in).

"FontWeight"
Peso de un carácter. Poniendo esta propiedad en "Bold" hace que MATLAB use una versión "negrita" de la fuente, cuando está disponible en el sistema.

"ForegroundColor"
Color de texto. Esta propiedad determina el color del texto definido por la propiedad "String". Especifica un color usando un vector de tres elementos RGB o un nombre predefinido en MATLAB.

REFERENCIAS

Referencias

Aubanell, A., Benseny, A., Delshams, A. (1993). *Útiles básicos de cálculo numérico*. Editorial Labor. Universitat Autónoma de Barcelona & Servei de Publicacions.

Bartle, R. G. (1976). *The elements of real analysis* (2d ed). Wiley.

Chapra, S. (2017). *Applied Numerical Methods with MATLAB, for Engineers and Scientists*. McGraw-Hill Education.

Chapra, S. C, & Canale, R. P. (2005). *Métodos numéricos para ingenieros: Con programas de aplicación*. McGraw-Hill.

Chapra, Steven C, Canale, R. P., & ProQuest. (2007). *Métodos numéricos para ingenieros* (5a. Ed.). McGraw-Hill Interamericana.

Deuflhard, P. (2004). *Newton methods for nonlinear problems: Affine invariance and adaptive algorithms*. Springer.

Díaz Moreno, J. M., Benítez Trujillo, F. (1998). *Introducción a los métodos numéricos para la resolución de ecuaciones*. Universidad de Cádiz, Servicio de Publicaciones.

Duque-Vega, P. R., & Gracia-Fadrique, J. (2015). Van der Waals, más que una ecuación cúbica de estado. *Educación Química*, 26(3), 187-194. https://doi.org/10.1016/j.eq.2015.05.003

García Merayo, F., & Nevot Luna, A. (1997). *Métodos numéricos: En forma de ejercicios resueltos*. Universidad Pontificia Comillas.

Grau Sánchez, M., & Noguera Batlle, M. (2001). *Cálculo numérico: Teoría y práctica*. Edicions UPC.

Jorquera, H., & Gelmi Weston, C. A. (2014). *Métodos numéricos aplicados a ingeniería: Casos de estudio en ingeniería de procesos usando MATLAB*. Pontificia Universidad Católica de Chile.

Kelley, C. T. (2003). *Solving nonlinear equations with Newton's method*. Society for Industrial and Applied Mathematics.

Kharab, A., & Guenther, R. B. (2006). *An introduction to numerical methods: A MATLAB approach* (2nd ed). Chapman & Hall/CRC.

Lablanquie, P., Aoto, T., Hikosaka, Y., Morioka, Y., Penent, F., & Ito, K. (2007). *Appearance of interatomic Coulombic decay in Ar, Kr, and Xe homonuclear dimers*. The Journal of Chemical Physics, 127(15), 154323. https://doi.org/10.1063/1.2778430

Martín Llorente, I., & Pérez García, V. M. (2007). *Cálculo numérico para computación en ciencia e ingeniería: Desarrollo práctico con MATLAB*. Síntesis.

Neuhauser, C., & Torres Suárez, A. (2011). *Matemáticas para ciencias*. Pearson.

Ortega, J. M., & Rheinboldt, W. C. (2000). *Iterative solution of nonlinear equations in several variables*. Society for Industrial and Applied Mathematics.

Pino, E., Valle, A., Condori, F., Mejía, J., Chávarri, E., & Alfaro, L. (2017). *Diseño Óptimo de Redes de Distribución de Agua Usando Un Software Basado En Microalgoritmos Genéticos Multiobjetivos*. Ribagua, Revista Iberoamericana del Agua, 4(1), 6-23. https://doi.org/10.1080/23863781.2017.1317087

Plaat, O. (1974). *Ecuaciones diferenciales ordinarias*. Reverté.

Süli, E., & Mayers, D. F. (2003). *An introduction to numerical analysis*. Cambridge University Press.

UNJBG

Fondo Editorial