Taller de Simulación Numérica
Curso 2021-22

Tema 4. Problemas de contorno 1D no lineales y evolutivos

F. Pena; P. Quintela

Consideremos el problema de contorno con CC generales:

$$\left\{ \begin{array}{l} y''(x) = F(x, y(x), y'(x)), x \in [a, b] \\ F(x, y(x), y'(x)) = u(x) + v(x)y^m(x) + w(x)y'(x), \\ -p_a y'(a) + h_a y(a) = y_a; \;\; p_b y'(b) + h_b y(b) = y_b. \end{array} \right. \qquad (P_{NL})$$

Supongamos que existe una única solución, $y(x) \in C^2([a, b])$, del problema (P_{NL}). Estudiaremos dos algoritmos distintos para la aproximación de su solución:

1. El algoritmo de iteración funcional.
2. El algoritmo de Newton.

$$y''(x) = u(x) + v(x)y^m(x) + w(x)y'(x), x \in [a, b], \ \& \ \mathrm{CC},$$

donde CC indica las Condiciones de Contorno (o bc *Boundary Conditions*) consideradas.
El objetivo es calcular un punto fijo del operador :

$$\begin{array}{rcl} \mathcal{F} : C^2[a, b] & \to & C^2[a, b] \\ \widehat{y} & \to & \mathcal{F}(\widehat{y}) = \widetilde{y}, \end{array}$$

siendo $\tilde{y}$ la solución del problema lineal:

$$\widetilde{y}''(x) = u(x) + v(x)\widehat{y}^{m-1}(x)\widetilde{y}(x) + w(x)\widetilde{y}'(x), x \in [a, b], \ \& \ \mathrm{CC}.$$

Obsérvese que el punto fijo de $\mathcal{F}$, verifica $\mathcal{F}(\widehat{y}) = \widehat{y}$, y, por tanto, $\widehat{y}(x)$ sería también solución del problema (P_{NL}), con lo cual $y(x) = \widehat{y}(x)$ por la unicidad de solución.

Sea $F_{\widehat{y}}\left(x,\widetilde{y},z\right)$ la función definida en $R=[a,b]\times\mathbb{R}\times\mathbb{R}$:

$$F_{\widehat{y}}\left(x,\widetilde{y},z\right)=u(x)+v(x)\widehat{y}^{m-1}(x)\widetilde{y}+w(x)z,$$

correspondiente al problema de contorno lineal:

$$\left\{\begin{array}{l}\widetilde{y}''(x)=F_{\widehat{y}}\left(x,\widetilde{y},\widetilde{y}'\right)=u(x)+\tilde{v}(x)\widetilde{y}\left(x\right)+w(x)\widetilde{y}'(x), x\in[a,b]\\ -p_a'\tilde{y}'(a)+h_a\tilde{y}(a)=y_a;\ p_b'\tilde{y}'(b)+h_b\tilde{y}(b)=y_b,\end{array}\right. \tag{6}$$

siendo $\tilde{v}(x)=v(x)\widehat{y}^{m-1}(x)$. Si $F_{\widehat{y}}$ y las CC verifican el resultado de existencia, el operador $\mathcal{F}$ estará bien definido. Así, en particular, si $v(x)>0$, $\tilde{v}(x)=v(x)\widehat{y}^{m-1}(x)>0$ si m es entero e impar. En otro caso, habrá que verificar que $\tilde{v}(x)$ toma valores positivos para garantizar la existencia (recordar que el resultado de existencia da condiciones suficientes, no necesarias).

$$y''(x) = u(x) + v(x)y^m(x) + w(x)y'(x), x \in [a, b], \text{ \& CC},$$

$$\begin{array}{rcl} \mathcal{F} : C^2[a, b] & \to & C^2[a, b] \\ \widehat{y} & \to & \mathcal{F}(\widehat{y}) = \widetilde{y}, \end{array}$$

Algoritmo de iteración funcional

- Sea $y_0 \in C^2[a, b]$ el iterante inicial.Conocido el iterante $y_k \in C^2[a, b]$, se construye $y_{k+1} = \mathcal{F}(y_k) \in C^2[a, b]$, $k \geq 0$, resolviendo:

$$\left\{ \begin{array}{l} y''_{k+1}(x) = F_{y_k}\left(x, y_{k+1}, y'_{k+1}\right) \\ \qquad = u(x) + \tilde{v}_k(x)y_{k+1}(x) + w(x)y'_{k+1}(x), x \in [a, b] \\ -p_a y'_{k+1}(a) + h_a y_{k+1}(a) = y_a; \;\; p_b y'_{k+1}(b) + h_b y_{k+1}(b) = y_b, \\ \tilde{v}_k(x) = v(x)y_k^{m-1}(x). \end{array} \right. \qquad (P_{NLIF})$$

Algoritmo de iteración funcional

- Sea $y_0 \in C^2[a,b]$ el iterante inicial. Conocido el iterante $y_k \in C^2[a,b]$, se construye $y_{k+1} = \mathcal{F}(y_k) \in C^2[a,b]$, $k \geq 0$, resolviendo:

$$\left\{ \begin{array}{l} y''_{k+1}(x) = u(x) + \tilde{v}_k(x) y_{k+1}(x) + w(x) y'_{k+1}(x), x \in [a,b] \\ -p_a y'_{k+1}(a) + h_a y_{k+1}(a) = y_a; \;\; p_b y'_{k+1}(b) + h_b y_{k+1}(b) = y_b, \\ \tilde{v}_k(x) = v(x) y_k^{m-1}(x). \end{array} \right. \qquad (P_{NLIF})$$

- Si $||y_{k+1} - y_k|| \leq \varepsilon$ y/o $||y''_{k+1}(x) - u(x) - v(x) y^m_{k+1}(x) - w(x) y'_{k+1}(x)|| < \varepsilon$ entonces:
    - Hay convergencia: $y(x) \simeq y_{k+1}(x)$;
- En otro caso se sigue iterando:
    - $k = k+1; y_k = y_{k+1};$
- end

4.1.2 Práctica 3. Implementación método iteración funcional

3.1 Elaborar la función MATLAB dfif.m para resolver el problema (P_{NLIF}) con CC generales. Considerar como argumentos de entrada:

- La estructura F, cuyos campos sean tres funciones anónimas 'u', 'v', 'w' con los coeficientes $u(x)$, $v(x)$ y $w(x)$, y el exponente m.
- La estructura bc, con campos 'blk_a', 'a', 'ya', 'ha', 'blk-b', 'b', 'yb', 'hb', indicando si la condición es Dirichlet o no en cada extremo, los extremos del intervalo, y los datos de cada CC.
- El escalar n indicando el número de pasos de discretización.
- La estructura nl, que define el iterante inicial y_0, el número maximo de iteraciones, $maxit$, y la tolerancia de convergencia, tol.
- Y como argumentos de salida:
- El vector x de puntos de discretización de la variable independiente.
- El vector y conteniendo la solución aproximada en los correspondientes puntos de discretización.
- El número de iteraciones realizadas, k.

```
function [x, y, k] = dfif(F, bc, n, nl)
% dfif Metodo de diferencias finitas para una EDO no lineal.
% Algoritmo de Iteración Funcional.
% [x,y,k] = dfif(F,bc,n,nl) aproxima la solucion en [bc.a, bc.b] en n pasos:
%y'' = F.u + F.v y^{F.m} + F.w y'
% y(bc.a) = bc.ya, si bc.blk_a es verdadero,
% -y'(bc.a) + bc.ha y(bc.a) = bc.ya, si bc.blk_a es falso,
% y(bc.b) = bc.yb, si bc.blk_b es verdadero,
% -y'(bc.b) + bc.hb y(bc.b) = bc.yb, si bc.blk_b es falso,
% El número máximo de iteraciones será nl.maxit
% El parámetro de tolerancia nl.tol
% El iterante inicial nl.yk
  Fl.u = F.u; Fl.v = @vlF; Fl.w = F.w;
  d1 = @(y,h) (y(3:end)-y(1:end-2))/(2*h); % Aproximación y'
  d2 = @(y,h) (y(3:end)-2*y(2:end-1)+y(1:end-2))/h^2; % Aprox. y''
```

```
% Iteraciones método iteración funcional
  for  k = 1:nl.maxit
     [x, y] = df(FI, bc, n);
     h = x(2) - x(1); pc = 2:length(x)-1;
     ydif = norm((y-nl.yk)./max(abs(y),eps), Inf);
     Fres = norm(d2(y,h) - F.u(x(pc)) - F.v(x(pc)).*y(pc).^F.m ...
     - F.w(x(pc)).*d1(y,h), Inf);
     if  ydif < nl.tol || Fres < nl.tol
        return
     end
        nl.yk = y;
    end
error('El algoritmo no converge.' )
% Función auxiliar del modelo linealizado en Iteración funcional
function   y = vIF(x)
% vIF Coeficiente
ṽ(x)  % para la EDO linealizada de iteracion funcional.
  y = F.v(x) .* nl.yk.^(F.m-1);
  end
```

3. 2 Diseñar un test académico que permita verificar el algoritmo de iteración funcional y su implementación.
Consideremos la función $y(x) = e^x + \cos(x)$, y consideremos $m \in \mathbb{R}$. Entonces, $y(x)$ es solución del problema no lineal:

$$\begin{aligned} y''(x) &= u(x) + y^m(x), x \in [0,1], \\ -y'(0) + y(0) &= 1;\; y'(1) + y(1) = 2e + \cos(1) - sen(1), \end{aligned}$$

siendo:

$$u(x) = -\left(e^x + \cos(x)\right)^m + e^x - \cos(x).$$

El operador de iteración funcional será:

$$\begin{array}{rcl} \mathcal{F} : C^2[0,1] & \rightarrow & C^2[0,1] \\ \widehat{y} & \rightarrow & F(\widehat{y}) = \widetilde{y}, \end{array}$$

con $\widetilde{y}$ la solución de:

$$\widetilde{y}''(x) = u(x) + \widehat{y}^{m-1}(x)\widetilde{y}(x), x \in [0,1], \ \& \text{ bc.}$$

La función

$$\tilde{v}_k(x) = v(x)y_k^{m-1}(x) = y_k^{m-1}(x).$$

- Calcular la solución aproximada utilizando el algoritmo de iteración funcional, con $h = 0{,}1$ y $h = 0{,}01$, y para $m = 1 : 0{,}4 : 3$.
- Representar gráficamente la solución exacta y las aproximadas, incluyendo una leyenda con las iteraciones utilizadas.
- Calcular el máximo error absoluto y el máximo error relativo.
- Calcular la norma $H^1(0,1)$ discreta del error absoluto.

3.3 Considerar el balance de masa de un reactor en estado estacionario y con decaimiento no lineal:

$$\begin{cases} c''(x) = \frac{\gamma}{D} c^m(x) + \frac{U}{D} c'(x), \ 0 < x < L, \\ -c'(0) + \frac{U}{D} c(0) = \frac{U}{D} c_{in}. \\ c'(L) = 0 \end{cases}$$

- Resuelve el problema anterior mediante diferencias finitas y el algoritmo de iteración funcional.
- Aplicar el algoritmo diseñado a la resolución del modelo para los datos: $D = 1m^2/h$, $U = 1m/h$, $\gamma = 0{,}2$, $c_{in} = 100 mol/m^3$, $L = 10m$, $h = 0{,}125m$.
- Comparar gráficamente los resultados con los obtenidos para el decaimiento lineal.
- Analizar la convergencia del algoritmo de iteración funcional para distintos valores de m, y distintos iterantes iniciales.

$$y''(x) = u(x) + v(x)y^m(x) + w(x)y'(x), x \in [a, b], \ \& \ \text{CC}, \qquad (P_{NL})$$

donde CC indica que se verifican las Condiciones de Contorno consideradas.
El objetivo es calcular un cero del operador: :

$$\begin{array}{rcl} \mathcal{G} : C^2_{cc}[a, b] & \longrightarrow & C[a, b] \\ \widehat{y} & \longrightarrow & \mathcal{G}\left(\widehat{y}\right) = r, \end{array}$$

siendo $C^2_{cc}[a, b]$ el conjunto de funciones de clase 2, que verifican las CC, y siendo r el residuo:

$$r\left(x\right) = \hat{y}''(x) - u(x) - v(x)\widehat{y}^m(x) - w(x)\widehat{y}'(x), x \in [a, b], \ \& \ \text{CC}.$$

El operador $\mathcal{G}$ esta bien definido. Obsérvese que el cero de $\mathcal{G}$, verifica $\mathcal{G}\left(\widehat{y}\right) = 0$, y, por tanto, $\widehat{y}\left(x\right)$ sería también solución del problema (P_{NL}), con lo cual $y(x) = \widehat{y}(x)$ por la unicidad de solución.

Supongamos que existe $y(x) \in C^2_{cc}[a,b]$ solución del problema:

$$y''(x) = u(x) + v(x)y^m(x) + w(x)y'(x), x \in [a,b], \ \& \text{ CC.} \qquad (P_{NL})$$

El objetivo es calcular un cero del operador :

$$\begin{array}{rcl} \mathcal{G} : C^2_{cc}[a,b] & \rightarrow & C[a,b] \\ \widehat{y} & \rightarrow & \mathcal{G}(\widehat{y}) = r, \end{array}$$

Puesto que $y(x)$ debe verificar:

$$y \in C^2_{cc}[a,b] \text{ y } \mathcal{G}(y) = 0.$$

conocido el iterante $y_k \in C^2_{cc}[a,b]$, el objetivo es aproximar y :

$$0 = \mathcal{G}(y) = \mathcal{G}(y_k) + D\mathcal{G}(y_k)(y - y_k) + O\left((y - y_k)^2\right).$$

El algoritmo de Newton aproxima el objetivo por su aproximación lineal de orden 1 . Se construye y_{k+1} tal que:

$$0 = \mathcal{G}(y_k) + D\mathcal{G}(y_k)(y_{k+1} - y_k), k \geq 0.$$

Cálculo de la $D\mathcal{G}\left(\hat{y}\right)$

- $$\mathcal{G}\left(\hat{y}\right) = \hat{y}''(x) - u(x) - v(x)\widehat{y}^{m}(x) - w(x)\widehat{y}'(x).$$

- $$D\mathcal{G}\left(\hat{y}\right)\left(\delta\hat{y}\right)(x) = \delta\hat{y}''(x) - mv(x)\hat{y}^{m-1}(x)\delta\hat{y}(x) - w(x)\delta\hat{y}'(x).$$

- Obtención del iterante $k+1$

$$0 = \mathcal{G}\left(y_k\right) + D\mathcal{G}\left(y_k\right)\left(y_{k+1} - y_k\right)$$
$$\Downarrow$$
$$y_k''(x) - u(x) - v(x)y_k^m(x) - w(x)y_k'(x) +$$
$$\left(y_{k+1} - y_k\right)''(x) - mv(x)y_k^{m-1}(x)\left(y_{k+1} - y_k\right)(x)$$
$$-w(x)\left(y_{k+1} - y_k\right)'(x) = 0.$$

Obtención del iterante $k+1$

- $$y_k''(x) - u(x) - v(x)y_k^m(x) - w(x)y_k'(x) +$$
$$(y_{k+1} - y_k)''(x) - mv(x)y_k^{m-1}(x)(y_{k+1} - y_k)(x)$$
$$-w(x)(y_{k+1} - y_k)'(x) = 0.$$

Agrupando términos:

$$y_{k+1}''(x) = u(x) + (1-m)v(x)y_k^m(x) + mv(x)y_k^{m-1}(x)y_{k+1}(x) + w(x)y_{k+}'$$

- Así, en cada iteración se resuelve un problema lineal con $\tilde{u}_k(x) = u(x) + (1-m)v(x)y_k^m(x)$, $\tilde{v}_k(x) = mv(x)y_k^{m-1}(x)$, imponiendo las CC del problema de partida.

Algoritmo de Newton

- Sea $y_0 \in C^2[a,b]$ el iterante inicial. Conocido el iterante $y_k \in C^2_{cc}[a,b]$, se construye $y_{k+1} \in C^2_{cc}[a,b]$, $k \geq 0$, resolviendo:

$$\left\{ \begin{array}{l} y''_{k+1}(x) = \tilde{u}_k(x) + \tilde{v}_k(x)\, y_{k+1}(x) + w(x) y'_{k+1}(x), x \in [a,b] \\ -p_a y'_{k+1}(a) + h_a y_{k+1}(a) = y_a; \;\; p_b y'_{k+1}(b) + h_b y_{k+1}(b) = y_b, \\ \tilde{u}_k(x) = u(x) + (1-m)\, v(x) y_k^m(x); \;\; \tilde{v}_k(x) = m v(x) y_k^{m-1}(x). \end{array} \right. \qquad (P_{NLIF})$$

- Si $||y_{k+1} - y_k|| \leq \varepsilon$ y/o $||y''_{k+1}(x) - u(x) - v(x) y^m_{k+1}(x) - w(x) y'_{k+1}(x)|| < \varepsilon$ entonces:
  - Hay convergencia: $y(x) \simeq y_{k+1}(x)$;
- En otro caso se sigue iterando:
  - $k = k+1; y_k = y_{k+1};$
- end

4.1.6 Práctica 3. Implementación del método de Newton

3.4 Elaborar la función MATLAB dfn.m para resolver el problema (P_{NLIF}) con CC generales. Considerar como argumentos de entrada:

- La estructura F, cuyos campos sean tres funciones anónimas 'u', 'v', 'w' con los coeficientes $u(x)$, $v(x)$ y $w(x)$, y el exponente m.
- La estructura bc, con campos 'blk_a', 'a', 'ya', 'ha', 'blk-b', 'b', 'yb', 'hb', indicando si la condición es Dirichlet o no en cada extremo, los extremos del intervalo, y los datos de cada CC.
- El escalar n indicando el número de pasos de discretización.
- La estructura nl, que define el iterante inicial y_0, el número maximo de iteraciones, $maxit$, y la tolerancia de convergencia, tol.

1. Y como argumentos de salida:
   - El vector x de puntos de discretización de la variable independiente.
   - El vector y conteniendo la solución aproximada en los correspondientes puntos de discretización.
   - El número de iteraciones realizadas, k.

3.5 Diseñar un test académico que permita verificar el algoritmo de Newton propuesto y su implementación.
Consideremos la función $y(x) = e^x + \cos(x)$, y consideremos $m \in \mathbb{R}$, $m > 1$. Entonces, $y(x)$ es solución del problema no lineal:

$$y''(x) = u(x) + y^m(x), x \in [0, 1], \text{ \& bc},$$

siendo:

$$u(x) = -\left(e^x + \cos(x)\right)^m + e^x - \cos(x).$$

En cada iteración de Newton deberá resolverse:

$$\begin{aligned} y''_{k+1}(x) &= u(x) + (1-m)\, v(x) y_k^m(t) + m v(x) y_k^{m-1}(x) y_{k+1}(x) \\ &\quad + w(x) y'_{k+1}(x) \\ &= u(x) + (1-m)\, y_k^m(t) + m y_k^{m-1}(x) y_{k+1}(x). \end{aligned}$$

- Así, en cada iteración se resuelve un problema lineal con $\tilde{u}_k(x) = u(x) + (1-m)\, y_k^m(t)$; $\tilde{v}_k(x) = m y_k^{m-1}(x)$, imponiendo las CC del problema de partida.

- Calcular la solución aproximada utilizando el algoritmo de Newton, con $h = 0{,}1$ y $h = 0{,}01$, y para $m = 1:1:10$.
- Representar gráficamente la solución exacta y las aproximadas, incluyendo una leyenda con las iteraciones utilizadas.
- Calcular el máximo error absoluto y el máximo error relativo.
- Calcular la norma $H^1(0,1)$ discreta del error absoluto.

3.6 Considerar el balance de masa de un reactor en estado estacionario y con decaimiento no lineal:

$$\begin{cases} c''(x) = \frac{\gamma}{D} c^m + \frac{U}{D} c'(x), \ 0 < x < L, \\ -c'(0) + \frac{U}{D} c(0) = \frac{U}{D} c_{in}. \\ c'(L) = 0 \end{cases}$$

- Resuelve el problema anterior mediante diferencias finitas y el algoritmo de Newton.
- Aplicar el algoritmo diseñado a la resolución del modelo para los datos: $D = 1 m^2/h$, $U = 1 m/h$, $\gamma = 0{,}2$, $c_{in} = 100 mol/m^3$, $L = 10m$, $n = 100$.
- Comparar gráficamente los resultados con los obtenidos para el decaimiento lineal.
- Analizar la convergencia del algoritmo de Newton para distintos valores de m, y distintos iterantes iniciales.

Consideremos el problema de contorno evolutivo:

$$\left\{ \begin{array}{l} l(x,t)\frac{\partial y(x,t)}{\partial t} - \frac{\partial^2 y(x,t)}{\partial x^2} + F\left(x, t, y(x,t), \frac{\partial y(x,t)}{\partial x}\right) = 0, \\ a \leq x \leq b,\, t_0 \leq t \leq t_f, \\ -\frac{\partial y(a,t)}{\partial x} + h_a y(a,t) = y_a(t),\ \ t_0 \leq t \leq t_f, \\ \frac{\partial y(b,t)}{\partial x} + h_b y(b,t) = y_b(t),\ \ t_0 \leq t \leq t_f, \\ y(x,t_0) = y_0(x),\ \ a \leq x \leq b \end{array} \right. \qquad (P_E)$$

Discretización en tiempo: Euler implícito

$$t^0 < t^1 < \ldots < t^{n_t} = t_f$$

siendo:

$$t^j = t_0 + j\Delta t, 0 \leq j \leq n_t,\ \ \Delta t = \frac{t_f - t_0}{n_t}.$$

$$
\left\{
\begin{array}{l}
l\left(x,t\right)\frac{\partial y(x,t)}{\partial t}-\frac{\partial^2 y(x,t)}{\partial x^2}+F\left(x,t,y\left(x,t\right),\frac{\partial y(x,t)}{\partial x}\right)=0,\\
a\leq x\leq b,\, t_0\leq t\leq t_f,\\
-\frac{\partial y(a,t)}{\partial x}+h_a y(a,t)=y_a\left(t\right),\ \ t_0\leq t\leq t_f,\\
\frac{\partial y(b,t)}{\partial x}+h_b y(b,t)=y_b\left(t\right),\ \ t_0\leq t\leq t_f,\\
y\left(x,t_0\right)=y_0\left(x\right),\ \ a\leq x\leq b.
\end{array}
\right. \qquad (P_E)
$$

$$
\left\{
\begin{array}{l}
l^j\left(x\right)\frac{y^j(x)-y^{j-1}(x)}{\Delta t}-\frac{d^2 y^j(x)}{dx^2}+F\left(x,t^j,y^j\left(x\right),\frac{dy^j(x)}{dx}\right)=0,\\
a\leq x\leq b, 1\leq j\leq n_t,\\
-\frac{dy^j(a)}{dx}+h_a y^j(a)=y_a^j,\ 1\leq j\leq n_t,\\
\frac{dy^j(b)}{dx}+h_b y^j(b)=y_b^j,\ 1\leq j\leq n_t,\\
y^0\left(x\right)=y_0\left(x\right),\ \ a\leq x\leq b,
\end{array}
\right. \qquad (P_{EDt})
$$

siendo $y^j\left(x\right)\simeq y\left(x,t^j\right)$, y dada una función $f\left(x,t\right)$ conocida, $f^j\left(x\right)=f\left(x,t^j\right)$.

$$
\begin{cases}
l^j(x)\,\frac{y^j(x)-y^{j-1}(x)}{\Delta t} - \frac{d^2y^j(x)}{dx^2} + F\left(x, t^j, y^j(x), \frac{dy^j(x)}{dx}\right) = 0, \\
a \leq x \leq b, 1 \leq j \leq n_t.
\end{cases}
\qquad (P_{EDt})
$$

$$
\frac{d^2y^j(x)}{dx^2} = l^j(x)\,\frac{y^j(x)-y^{j-1}(x)}{\Delta t} + F\left(x, t^j, y^j(x), \frac{dy^j(x)}{dx}\right).
$$

Discretización en espacio: Diferencias finitas

$$
a = x_0 < x_1 < \ldots < x_{n_x} = b,
$$

siendo:

$$
x_i = a + ih,\ \ 0 \leq i \leq n_x,\ \ h = \frac{b-a}{n_x}.
$$

$$\frac{d^2 y^j(x)}{dx^2} = l^j(x)\,\frac{y^j(x) - y^{j-1}(x)}{\Delta t} + F\left(x, t^j, y^j(x), \frac{dy^j(x)}{dx}\right).$$

Discretización en espacio: Diferencias finitas

$$a = x_0 < x_1 < \ldots < x_{n_x} = b,$$

siendo:

$$x_i = a + ih,\ 0 \le i \le n_x,\ h = \frac{b-a}{n_x}.$$

$$\left\{\begin{array}{l} \frac{y_{i+1}^j - 2y_i^j + y_{i-1}^j}{h^2} = l_i^j \frac{y_i^j - y_i^{j-1}}{\Delta t} + F(x_i, t^j, y_i^j, \frac{y_{i+1}^j - y_{i-1}^j}{2h}), \\ 1 \le i \le n_x - 1, 1 \le j \le n_t, \end{array}\right. \qquad (P_{EDtx})$$

siendo: $y_i^j \simeq y\left(x_i, t^j\right)$.

$$\frac{d^2 y^j(x)}{dx^2} = l^j(x)\,\frac{y^j(x) - y^{j-1}(x)}{\Delta t} + F\left(x, t^j, y^j(x), \frac{dy^j(x)}{dx}\right).$$

Si consideramos el caso lineal $(m = 1)$:

$$F(x, t, y, \frac{\partial y}{\partial x}) = u(x,t) + v(x,t)\,y(x,t) + w(x,t)\,\frac{\partial y}{\partial x}(x,t),$$

entonces el problema lineal discretizado en tiempo (P_{ELDt}) es:

$$\begin{aligned}\frac{d^2 y^j(x)}{dx^2} &= l^j(x)\,\frac{y^j(x) - y^{j-1}(x)}{\Delta t} + u^j(x)\\ &\quad + v^j(x)\,y^j(x) + w^j(x)\,\frac{dy^j(x)}{dx}. \qquad (P_{ELDt})\end{aligned}$$

$$
\begin{aligned}
\frac{d^2 y^j(x)}{dx^2} &= l^j(x)\,\frac{y^j(x) - y^{j-1}(x)}{\Delta t} + u^j(x) \\
&\quad + v^j(x)\,y^j(x) + w^j(x)\,\frac{dy^j(x)}{dx} \\
&= \underbrace{\left(u^j(x) - \frac{l^j(x)\,y^{j-1}(x)}{\Delta t}\right)}_{\tilde{u}^j(x)} \\
&\quad + \underbrace{\left(v^j(x) + \frac{l^j(x)}{\Delta t}\right)}_{\tilde{v}^j(x)} y^j(x) + w^j(x)\,\frac{dy^j(x)}{dx} \\
&= \tilde{F}^j(x, t, y, \frac{dy}{dx}). \qquad (P_{ELDt})
\end{aligned}
$$

$$
\begin{aligned}
\frac{d^2 y^j(x)}{dx^2} &= \underbrace{\left(u^j(x) - \frac{l^j(x)\, y^{j-1}(x)}{\Delta t} \right)}_{\tilde{u}^j(x)} \\
&\quad + \underbrace{\left(v^j(x) + \frac{l^j(x)}{\Delta t} \right)}_{\tilde{v}^j(x)} y^j(x) + w^j(x) \frac{dy^j(x)}{dx} \\
&= \tilde{F}^j(x, t, y^j, \frac{dy^j}{dx}). \qquad (P_{ELDt})
\end{aligned}
$$

$$
\begin{array}{l}
-\frac{dy^j(a)}{dx} + h_a y^j(a) = y_a^j,\ 1 \le j \le n_t, \\
\frac{dy^j(b)}{dx} + h_b y^j(b) = y_b^j,\ 1 \le j \le n_t.
\end{array}
$$

$$\frac{d^2 y^j(x)}{dx^2} = \tilde{u}^j(x) + \tilde{v}^j(x)\, y^j(x) + w^j(x)\,\frac{dy^j(x)}{dx}.$$

$$\left\{\begin{array}{l} \frac{y_{i+1}^j - 2y_i^j + y_{i-1}^j}{h^2} = l_i^j \frac{y_i^j - y_i^{j-1}}{\Delta t} + F(x_i, t^j, y_i^j, \frac{y_{i+1}^j - y_{i-1}^j}{2h}) \\ \quad = \tilde{u}_i^j + \tilde{v}_i^j y_i^j + w_i^j \frac{y_{i+1}^j - y_{i-1}^j}{2h} \\ 1 \leq i \leq n_x - 1, 1 \leq j \leq n_t, \end{array}\right. \qquad (P_{ELDtx})$$

Multiplicando por h^2 y agrupando términos:

$$\begin{aligned} & -(1 + \frac{h}{2} w_i^j) y_{i-1}^j + (2 + h^2 v_i^j + \frac{h^2}{\Delta t} l_i^j) y_i^j + (-1 + \frac{h}{2} w_i^j) y_{i+1}^j \\ = \; & -h^2 u_i^j + \frac{h^2}{\Delta t} l_i^j y_i^{j-1},\ 1 \leq i \leq n_x - 1, 1 \leq j \leq n_t. \end{aligned}$$

Para cada $j, 1 \leq j \leq n_t$ se resuelve:

$$
\begin{aligned}
& -(1+\frac{h}{2}w_i^j)y_{i-1}^j + (2+h^2 v_i^j + \frac{h^2}{\Delta t} l_i^j)y_i^j + (-1+\frac{h}{2}w_i^j)y_{i+1}^j \\
= \quad & -h^2 u_i^j + \frac{h^2}{\Delta t} l_i^j y_i^{j-1},\ 1 \leq i \leq n_x - 1, 1 \leq j \leq n_t.
\end{aligned}
$$

- En cada paso de tiempo el sistema es tridiagonal de $n_x - 1$ ecuaciones, con $n_x + 1$ incógnitas:

$$a_i^j = -(1+\frac{h}{2}w_i^j);\ d_i^j = \left(2+h^2 v_i^j\right) + \frac{h^2}{\Delta t} l_i^j;$$

$$c_i^j = \left(-1+\frac{h}{2}w_i^j\right);\ b_i^j = -h^2 u_i^j + \frac{h^2}{\Delta t} l_i^j y_i^{j-1}.$$

$$a_i^j y_{i-1}^j + d_i^j y_i^j + c_i^j y_{i+1}^j = b_i^j,\ 1 \leq i \leq n_x - 1.$$

El sistema se completa con la discretización de las correspondientes condiciones de contorno.

4.2.1 Problemas de contorno 1D.

Modelos evolutivos. Caso lineal. Condiciones de contorno

Discretización en $x_0 = a$:

$$\frac{d^2 y^j(x)}{dx^2} = \tilde{u}^j(x) + \tilde{v}^j(x)\, y^j(x) + w^j(x)\,\frac{dy^j(x)}{dx},$$

$$-\frac{dy^j(a)}{dx} + h_a y^j(a) = y_a^j,\ 1 \leq j \leq n_t.$$

La discretización obtenida fue:

$$(1 + hh_a + \frac{h^2}{2}(\tilde{v}_0^j + h_a w_0^j))y_0^j - y_1^j = hy_a^j - \frac{h^2}{2}(\tilde{u}_0^j - w_0^j y_a^j).$$

$$\overbrace{(1 + hh_a + \frac{h^2}{2}(v_0^j + \frac{l_0^j}{\Delta t} + h_a w_0^j))}^{d_0^j}\, y_0^j - y_1^j$$

$$= \underbrace{hy_a^j - \frac{h^2}{2}(u_0^j - \frac{l_0^j y_0^{j-1}}{\Delta t} - w_0^j y_a^j)}\,.$$

Discretización en $x_{n_x} = b$:

$$\frac{d^2 y^j(x)}{dx^2} = \tilde{u}^j(x) + \tilde{v}^j(x)\, y^j(x) + w^j(x)\, \frac{dy^j(x)}{dx},$$

$$\frac{dy^j(b)}{dx} + h_b y^j(b) = y_b^j,\ 1 \leq j \leq n_t.$$

La discretización obtenida fue:

$$-y^j_{n_x-1} + \left(1 + hh_b + \frac{h^2}{2}\left(\tilde{v}^j_{n_x} - h_b w^j_{n_x}\right)\right) y_{n_x} = hy_b^j - \frac{h^2}{2}\left(\tilde{u}^j_{n_x} + w^j_{n_x} y_b^j\right).$$

$$-y^j_{n_x-1} + \overbrace{\left(1 + hh_b + \frac{h^2}{2}\left(v^j_{n_x} + \frac{l^j_{n_x}}{\Delta t} - h_b w^j_{n_x}\right)\right)}^{d^j_{n_x}} y^j_{n_x}$$

$$= \quad hy_b^j - \frac{h^2}{2}\left(u^j_{n_x} - \frac{l^j_{n_x} y^{j-1}_{n_x}}{\Delta t} + w^j_{n_x} y_b^j\right).$$

Para cada j, $1 \leq j \leq n_t$, se resuelve el sistema algebraico:

$$\left\{\begin{array}{l} d_0^j y_0^j - y_1^j = b_0^j \\ a_i^j y_{i-1}^j + d_i^j y_i^j + c_i^j y_{i+1}^j = b_i^j,\ 1 \leq i \leq n_x - 1, \\ -y_{n_x-1}^j + d_{n_x} y_{n_x}^j = b_{n_x}^j, \end{array}\right.$$

siendo:

$$\begin{aligned} d_0^j &= (1 + hh_a + \frac{h^2}{2}(v_0^j + \frac{l_0^j}{\Delta t} + h_a w_0^j)), \\ b_0^j &= hy_a^j - \frac{h^2}{2}(u_0^j - \frac{l_0^j y_0^{j-1}}{\Delta t} - w_0^j y_a^j), \\ d_{n_x}^j &= (1 + hh_b + \frac{h^2}{2}(v_{n_x}^j + \frac{l_{n_x}^j}{\Delta t} - h_b w_{n_x}^j)), \\ b_{n_x}^j &= hy_b^j - \frac{h^2}{2}(u_{n_x}^j - \frac{l_{n_x}^j y_{n_x}^{j-1}}{\Delta t} + w_{n_x}^j y_b^j). \end{aligned}$$

- Ecuación en diferencias:

$$a_i^j y_{i-1}^j + d_i^j y_i^j + c_i^j y_{i+1}^j = b_i^j,\ 1 \leq i \leq n_x - 1,$$

- Si l y v son funciones positivas, entonces la matriz del sistema es de diagonal estrictamente dominante:

$$\begin{aligned} |d_i^j| &= 2 + h^2 v_i^j + \frac{h^2}{\Delta t} l_i^j > 2, \\ |a_i^j| + |c_i^j| &= |-(1 + \frac{h}{2} w_i^j)| + |\left(-1 + \frac{h}{2} w_i^j\right)| = 2, \end{aligned}$$

si h se toma suficientemente pequeño para que $|\frac{h}{2} w_i^j| < 1$ (basta tomar $h < \frac{2}{M_w}$) siendo M_w la cota superior de la función $w\,(x,t)$. análogamente, se puede comprobar que se mantiene la diagonal estrictamente dominante en la primera y última fila de la matriz.

Estabilidad de la discretización propuesta:

Relación entre Δt y h. Si se observa la $i-$ésima ecuación en (P_{ELDtx}), el coeficiente de y_i^j en la derivada respecto al tiempo es $\frac{l_i^j}{\Delta t}$, coeficiente que debe prevalecer sobre los asociados a la discretización en espacio, $-\frac{2}{h^2} - v_i^j$, para garantizar que la evolución en tiempo esté bien considerada. Así, en cada tiempo t^j :

$$\frac{l_i^j}{\Delta t} > |-\frac{2}{h^2} - v_i^j| = |\frac{2}{h^2} + v_i^j|, 1 \leq i \leq n_x - 1, 1 \leq j \leq n_t.$$

Para garantizar esta condición es suficiente que:

$$\frac{m_l}{\Delta t} > \frac{2}{h^2} + M_v \Rightarrow \Delta t < \frac{h^2 m_l}{2 + M_v h^2},$$

siendo m_l y M_v el mínimo y el máximo de las funciones l y v, respectivamente. Esta condición puede demostrase de forma más rigurosa y se conoce como condición de Courant. Supone una seria restricción como se verá en prácticas.

Estabilidad de la discretización propuesta:

Restricción sobre h. Si se observa de nuevo la $i-$ésima ecuación de (P_{ELDtx}), debe garantizarse que el término que integra la derivada segunda respecto al espacio, $\frac{1}{h^2}$, prevalece sobre que el que incorpora la derivada primera, $\frac{w_i^j}{2h}$. Ello garantizará que ambas condiciones de contorno se puedan verificar suavemente. En particular, en cada tiempo t^j :

$$\frac{1}{h^2} > |\frac{w_i^j}{2h}|, 1 \leq i \leq n_x, 1 \leq j \leq n_t.$$

Para garantizar esta condición es suficiente que:

$$\frac{1}{h^2} > \frac{M_w}{2h}, \Rightarrow h < \frac{2}{M_w},$$

siendo M_w el máximo de la función w. Puede observarse que es la misma condición que la obtenida en la Práctica 2.

4.1 Elaborar la función MATLAB dfnle.m para resolver el problema (P_{NLE}) con CC generales. Considerar como argumentos de entrada:

- La estructura F, cuyos campos sean 4 funciones anónimas 'l(x,t)' 'u(x,t)', 'v(x,t)', 'w(x,t)' con los coeficientes $l(x,t)$, $u(x,t)$, $v(x,t)$ y $w(x,t)$, con argumentos vectores de discretización, x, t, y el exponente m.
- La estructura bi, que define las condiciones de contorno y la condición inicial. Tiene la misma estructura que la de los ejercicios anteriores, excepto:
-
    - bi.ya(t) y bi.yb(t) con argumento el vector de tiempos de discretización.
    - El intervalo temporal definido en [bi.t0, bi.tf].
    - La condición inicial bi.y0, que es un vector de la misma dimensión de x.
- El escalar n_x indicando el número de pasos de discretización en espacio.
- El escalar n_t indicando el número de pasos de discretización en tiempo.

- Y como argumentos de salida:
- El vector x de puntos de discretización de la variable espacial.
- El vector t de puntos de discretización de la variable temporal.
- La matriz $Y \in M_{(n_t+1)\times(n_x+1)}$ conteniendo la solución aproximada.

4.2 **Diseñar un test académico** con solución exacta $y(x) = te^x$, con $(x,t) \in [0,2] \times [1,5]$, que permita verificar el algoritmo propuesto en el Ejercicio 3.7 y su implementación. Considerar funciones l, v,y w en las condiciones del resultado de existencia y unicidad.

- Calcular el máximo error absoluto $||y(x,t) - y_{aprox}(x,t)||$, y analizar su evolución con respecto al tiempo: $||y\left(x,t^j\right) - y^j(x)||$, $1 \leq j \leq n_t$.
- Analizar la estabilidad del método para distintos valores de Δt y h.
- Escribir una función que permita calcular las normas $||L^\infty\left((t_0,t_f);L^2(a,b)\right)||$ y $||L^\infty\left((t_0,t_f);H^1(a,b)\right)||$ discretas para el error absoluto.
- Realizar la representación gráfica de la solución exacta frente a la aproximada para distintos valores de t.
- Representar gráficamente la solución aproximada en el intervalo $[0,2] \times [1,5]$.

Consideramos un **primer test lineal**:

$$
\begin{cases}
l(x,t)\frac{\partial y(x,t)}{\partial t} - \frac{\partial^2 y(x,t)}{\partial x^2} + F\left(x,t,y(x,t),\frac{\partial y(x,t)}{\partial x}\right) = 0,\\
0 \le x \le 2, 1 \le t \le 5,\\
y(x,1) = y_0(x),\ 0 \le x \le 2,\\
-\frac{\partial y(0,t)}{\partial x} + h_a y(0,t) = y_a(t),\ 1 \le t \le 5,\\
\frac{\partial y(2,t)}{\partial x} + h_b y(2,t) = y_b(t),\ 1 \le t \le 5.
\end{cases}
$$

Consideramos como **solución** $y(x,t) = te^x$. Tomemos $h_a = 1$, $h_b = 2$, $v(x,t) = \frac{1}{t}$, $w(x,t) = t$, $l(x,t) = 1$. En consecuencia:

$$
\begin{aligned}
F(x,t,y,\frac{\partial y}{\partial x}) &= u(x,t) + v(x,t)\,y(x,t) + w(x,t)\frac{\partial y}{\partial x}(x,t)\\
&= u(x,t) + \frac{1}{t}y(x,t) + t\frac{\partial y}{\partial x}(x,t).
\end{aligned}
$$

Cálculo de datos

Para el cálculo de $u(x,t)$ se calcula para $y(x,t) = te^x$:

$$
\left\{
\begin{array}{l}
\frac{\partial y(x,t)}{\partial t} = e^x;\ \frac{\partial^2 y(x,t)}{\partial x^2} = te^x \\
l(x,t)\frac{\partial y(x,t)}{\partial t} - \frac{\partial^2 y(x,t)}{\partial x^2} + f\left(x, t, y(x,t), \frac{\partial y(x,t)}{\partial x}\right) = 0 \Rightarrow \\
\\
\frac{\partial y(x,t)}{\partial t} - \frac{\partial^2 y(x,t)}{\partial x^2} + u(x,t) + \frac{1}{t} y(x,t) + t\frac{\partial y}{\partial x}(x,t) = 0 \Rightarrow \\
u(x,t) = \frac{\partial^2 y(x,t)}{\partial x^2} - \frac{\partial y(x,t)}{\partial t} - \frac{1}{t} y(x,t) - t\frac{\partial y}{\partial x}(x,t) \\
= te^x - e^x - \frac{1}{t} te^x - t^2 e^x = e^x\left(-t^2 + t - 2\right).
\end{array}
\right.
$$

Mientras que las condiciones de contorno serán:

$$
\begin{aligned}
-\frac{\partial y(0,t)}{\partial x} + y(0,t) &= y_a(t),\ 1 \leq t \leq 5 \Rightarrow y_a(t) = t - t = 0. \\
\frac{\partial y(2,t)}{\partial x} + 2y(2,t) &= y_b(t),\ 1 \leq t \leq 5 \Rightarrow y_b(t) = 3te^2.
\end{aligned}
$$

Así $y\left(x,t\right)=te^{x}$ es solución del problema de contorno evolutivo:

$$\left\{\begin{array}{l}\frac{\partial y(x,t)}{\partial t}-\frac{\partial^2 y(x,t)}{\partial x^2}+e^{x}\left(-t^2+t-2\right)+\frac{1}{t}y\left(x,t\right)+t\frac{\partial y}{\partial x}\left(x,t\right)=0,\\ x\in\left[0,2\right],t\in\left[1,5\right],\\ -\frac{\partial y(0,t)}{\partial x}+y(0,t)=0,\ 1\leq t\leq 5,\\ \frac{\partial y(2,t)}{\partial x}+2y(2,t)=3te^{2},\ 1\leq t\leq 5\\ y\left(x,1\right)=e^{x}.\end{array}\right.$$

$$
\begin{aligned}
\frac{d^2 y^j(x)}{dx^2} &= \underbrace{\left(u^j(x) - \frac{l^j(x)\, y^{j-1}(x)}{\Delta t}\right)}_{\tilde{u}^j(x)} \\
&\quad + \left(v^j(x)\left(y^j\right)^m(x) + \frac{l^j(x)}{\Delta t} y^j(x)\right) \\
&\quad + w^j(x)\,\frac{dy^j(x)}{dx},
\end{aligned}
$$

Algoritmo de iteración funcional

$$\frac{d^2y^j}{dx^2} = \tilde{u}^j + \left(v^j\left(y^j\right)^m + \frac{l^j}{\Delta t}y^j\right) + w^j\frac{dy^j}{dx}, 1 \leq j \leq n_t.$$

Algoritmo de iteración funcional: Conocido $\left(y^j\right)_k$, se construye

$$\left(y^j\right)_{k+1} = \tilde{\mathcal{F}}^j\left(\left(y^j\right)_k\right), \;\; k \geq 0,$$

siendo

$$\begin{array}{rcl} \tilde{\mathcal{F}}^j : C^2[a,b] & \rightarrow & C^2[a,b] \\ \widehat{y} & \rightarrow & \tilde{\mathcal{F}}^j\left(\widehat{y}\right) = \widetilde{y}, \text{ solución de} \end{array}$$

$$\widetilde{y}''(x) = \tilde{u}^j(x) + \left(v^j(x)\,\widehat{y}^{m-1}(x) + \frac{l^j(x)}{\Delta t}\right)\widetilde{y}(x)$$
$$+w^j(x)\widetilde{y}'(x), x \in [a,b], \;\& \text{ CC.}$$

Algoritmo de iteración funcional:

- Sea $\left(y^j\right)_0 \in C^2[a,b]$.
- Conocido el iterante $\left(y^j\right)_k \in C^2[a,b]$, se construye $\left(y^j\right)_{k+1} = \tilde{\mathcal{F}}^j\left(\left(y^j\right)_k\right)$, la solución de:

$$\left(y^j\right)''_{k+1}(x) = \tilde{u}^j(x) + \overbrace{\left(v^j(x)\left(y^j(x)\right)_k^{m-1} + \frac{l^j(x)}{\Delta t}\right)}^{\tilde{v}_k^j}\left(y^j\right)_{k+1}(x)$$
$$+w^j(x)\left(y^j\right)'_{k+1}(x), x \in [a,b], \ \& \text{ CC.}$$

- Si $\|\left(y^j\right)_{k+1} - \left(y^j\right)_k\| \leq \varepsilon$ y/o $\|\left(y^j\right)''_{k+1} - \left(\tilde{u}^j + v^j\left(\left(y^j\right)_{k+1}\right)^m + \frac{l^j}{\Delta t}\left(y^j\right)_{k+1} + w^j\left(y^j\right)'_{k+1}\right)\| < \varepsilon$ entonces
  - $y^j(x) \simeq \left(y^j\right)_{k+1}(x)$;
  - en otro caso
  - $k = k+1$;

$$\frac{d^2 y^j(x)}{dx^2} = \underbrace{\left(u^j(x) - \frac{l^j(x)\, y^{j-1}(x)}{\Delta t}\right)}_{\tilde{u}^j(x)} + \left(v^j(x)\left(y^j\right)^m(x) + \frac{l^j(x)}{\Delta t} y^j(x)\right) + w^j(x)\,\frac{dy^j(x)}{dx},$$

Algoritmo de Newton

Algoritmo de Newton: La solución es **un cero del operador**:

$$\begin{array}{rcl} \tilde{\mathcal{G}}^j : C^2_{cc}[a,b] & \to & C[a,b] \\ \widehat{y} & \to & \tilde{\mathcal{G}}^j(\widehat{y}) = \widetilde{y}, \text{ definido por} \end{array}$$

$$\widetilde{y}(x) = \widehat{y}''(x) - \tilde{u}^j(x) - v^j(x)\widehat{y}^m(x) - \frac{l^j(x)}{\Delta t}\widehat{y}(x)$$
$$- w^j(x)\widehat{y}'(x), x \in [a,b].$$

Cálculo de la $D\tilde{\mathcal{G}}^j(\widehat{y})$

$$\tilde{\mathcal{G}}^j\left(\widehat{y}\right) = \widehat{y}'' - \tilde{u}^j - v^j(\widehat{y})^m - \frac{l^j}{\Delta t}\widehat{y} - w^j\widehat{y}'.$$

$$D\tilde{\mathcal{G}}^j\left(\widehat{y}\right)(r) = r'' - \left(mv^j(\widehat{y})^{m-1} + \frac{l^j}{\Delta t}\right) r - w^j r'$$

$$0 = \tilde{\mathcal{G}}^j\left(\left(y^j\right)_k\right) + D\tilde{\mathcal{G}}^j\left(\left(y^j\right)_k\right)\left(\left(y^j\right)_{k+1} - \left(y^j\right)_k\right)$$

$$\Longrightarrow \left(y^j\right)_k'' - \tilde{u}^j - v^j\left(\left(y^j\right)_k\right)^m - \frac{l^j}{\Delta t}\left(y^j\right)_k - w^j\left(y^j\right)_k'$$

$$+\left(\left(y^j\right)_{k+1} - \left(y^j\right)_k\right)'' - \left(mv^j\left(\left(y^j\right)_k\right)^{m-1} + \frac{l^j}{\Delta t}\right)\left(\left(y^j\right)_{k+1} - \left(y^j\right)_k\right)$$

$$-w^j\left(\left(y^j\right)_{k+1} - \left(y^j\right)_k\right)' = 0.$$

Algoritmo de Newton:

$$\left(y^j\right)''_k - \tilde{u}^j - v^j \left(\left(y^j\right)_k\right)^m - \frac{l^j}{\Delta t}\left(y^j\right)_k - w^j \left(y^j\right)'_k$$
$$+ \left(\left(y^j\right)_{k+1} - \left(y^j\right)_k\right)'' - \left(m v^j \left(\left(y^j\right)_k\right)^{m-1} + \frac{l^j}{\Delta t}\right)\left(\left(y^j\right)_{k+1} - \left(y^j\right)_k\right)$$
$$-w^j \left(\left(y^j\right)_{k+1} - \left(y^j\right)_k\right)' = 0.$$

$$\begin{aligned}\left(y^j\right)''_{k+1}(x) \quad = \quad & \overbrace{\tilde{u}^j(x) + (1-m)\, v^j(x) \left(\left(y^j\right)_k\right)^m}^{\tilde{u}^j_k} \\ & + \underbrace{\left(m v^j(x) \left(\left(y^j\right)_k\right)^{m-1} + \frac{l^j(x)}{\Delta t}\right)}_{\tilde{v}^j_k}\left(y^j\right)_{k+1} \\ & + w^j(x) \left(y^j\right)'_{k+1}(x)\end{aligned}$$

Algoritmo de Newton:

- Sea $\left(y^j\right)_0 \in C^2[a,b]$.
- Conocido el iterante $\left(y^j\right)_k \in C^2[a,b]$, se construye $\left(y^j\right)_{k+1} \in C^2_{cc}[a,b]$, $k \geq 0$, como la solución del problema:

$$\left(y^j\right)''_{k+1}(x) = \tilde{u}^j_k(x) + \tilde{v}^j_k \left(y^j\right)_{k+1} + w^j(x) \left(y^j\right)'_{k+1}(x) \text{ \& CC}$$

- Si $|| \left(y^j\right)_{k+1} - \left(y^j\right)_k || \leq \varepsilon$ y/o $|| \left(y^j\right)''_{k+1} - \left(\tilde{u}^j + v^j \left(\left(y^j\right)_{k+1}\right)^m + \frac{l^j}{\Delta t} \left(y^j\right)_{k+1} + w^j \left(y^j\right)'_{k+1}\right) || < \varepsilon$ entonces
    - $y^j(x) \simeq \left(y^j\right)_{k+1}(x)$;
    - en otro caso
    - $k = k+1$;
    - end

4.4 Práctica 4. Aplicación a un reactor evolutivo y decaimiento no lineal

4.3

$$\left\{ \begin{array}{l} \frac{1}{D}\frac{\partial c}{\partial t} - \frac{\partial^2 c}{\partial x^2} + \frac{U}{D}\frac{\partial c}{\partial x} + \frac{\gamma}{D}c^m = 0,\ 0 < x < L, t \in [0, t_f], \\ -\frac{dc}{dx}(0,t) + \frac{U}{D}c(0,t) = \frac{U}{D}c_{in}, t \in [0, t_f], \\ \frac{\partial c}{\partial x}(L,t) = 0, t \in [0, t_f], \\ c(x,0) = 0,\ 0 < x < L, \end{array} \right.$$

siendo $m \in \mathbb{R}$, $m \geq 1$.

- Resuelve el problema anterior utilizando la función **dfnle.m** con decaimiento lineal y los valores $t_f = 25h$, $D = 2m^2/h$, $U = 2m/h$, $\gamma = 0{,}2$, $c_{in} = 100mol/m^3$, $L = 10m$.
- Representa gráficamente la solución aproximada para $t \in \{0{,}25,\ 2{,}5,\ 7{,}5,\ 25\}h$. Incluye también la gráfica del problema estacionario. Comparar el resultado con la gráfica B5.3a de [Caldwell].

4.4 Práctica 4. Aplicación a un reactor evolutivo y decaimiento no lineal

- Verificar numericamente la conservación de la masa en el reactor, representando gráficamente la norma $L^{\infty}(0, L)$ discreta del residuo de la EDO frente al tiempo.
- Cálculo la norma $L^{\infty}\left(0, t_f; H^1(0, L)\right)$ del residuo de la EDP.
- Aplicar la función **dfnle.m** al modelo con decaimiento no lineal y: $D = 1m^2/h$, $U = 1m/h$, $\gamma = 0{,}2$, $c_{in} = 100 mol/m^3$, $L = 10m$, $m = 2$, $h = 0{,}25m$.Representar la solución con contourf.