

UNIVERSITAT POLITÈCNICA DE CATALUNYA
ENGINYERIA QUÍMICA
MÈTODES NUMÈRICS
PRÀCTIQUES DE MATLAB 2^{on} QUADR. 2009–10

INSTRUCCIONS GENERALS:

- Les/els alumnes han de formar grups de tres persones per a fer les pràctiques. Els grups han de ser estables (durar tot el quadrimestre). Cada grup s'ha d'inscriure enviant un correu electrònic a l'adreça de l'assignatura `numeric@vilma.upc.edu`, que indiqui els integrants i un correu electrònic de contacte.
- Per entregar les pràctiques: envieu un correu des de l'adreça del vostre grup a la de l'assignatura, amb el fitxer de la pràctica adjuntat.
- La penalització per entregar tard una pràctica és de 1 punt en la seva nota (sobre 10) per cada dia de retard, els 5 primers dies després del termini.
- Les pràctiques corregides seran retornades per correu electrònic a l'adreça de cada grup.
- Només es permet una entrega de cada pràctica per grup, i la versió entregada només pot escriure per pantalla si les instruccions ho demanen explícitament.

PRÀCTIQUES COMUNES A TOTS ELS GRUPS:

Heu de fer les següents funcions per Matlab:

Pràctica 1: Projecció ortogonal

Termini d'entrega: 2010/2/26

Heu de fer una funció de MATLAB que calculi la projecció ortogonal d'un punt sobre una varietat lineal, en qualsevol dimensió, i la distància a la que es troben.

La varietat lineal V vindrà donada per les seves equacions implícites, és a dir que serà solució d'un sistema d'equacions lineals

$$A \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = b$$

La projecció ortogonal d'un punt P en la varietat V serà un punt amb coordenades $x = (x_1, x_2, \dots, x_n)$ que compleixi les condicions

- $Ax = b$ (aquesta indica que x és de la varietat V).
- $\langle x - P, v \rangle = 0$ per tot vector director v de la varietat V .

Si la varietat V ve donada per un sistema d'equacions implícites $Ax = b$, el seu subespai director és $\text{Nuc } A$, de manera que la condició d'ortogonalitat a tot vector director la podeu convertir en un sistema d'equacions lineals

$$C(x - P) = 0$$

o, equivalentment,

$$Cx = CP$$

on C és una matriu que té *per files* una base de $\text{Nuc } A$. Aleshores, el sistema d'equacions complet que calcula la projecció ortogonal s'obté reunint les equacions dels dos sistemes $Ax = b$ i $Cx = CP$. Un cop resolt, la distància buscada es la norma del vector $x - P$.

La vostra funció:

- Rebrà com a arguments la matriu A i el vector b que formen les equacions implícites de la varietat lineal V de \mathbb{R}^n , un punt $P \in \mathbb{R}^n$, i una distància de seguretat tol .
- Calcularà la projecció ortogonal de P sobre la subvarietat lineal V .
- Trobarà la distància entre P i la seva projecció ortogonal, que és també la distància entre P i la subvarietat lineal V .
- Si la distància de P a V és menor que tol escriurà per pantalla el text *Punt proper a la varietat*.
- Retornarà a l'usuari la projecció ortogonal de P en V , i la distància.

Nom del fitxer: projeccioimplicita.m

Capçalera de la funció: `function [projeccio,distancia]=projeccioimplicita(A,b,P,tol)`

INPUT: A =matriu de coeficients de les equacions de la varietat, b =terme independent de les equacions de la varietat, P =punt de l'espai ambient (en columna), tol =distància de seguretat.

OUTPUT: `projeccio`=projecció ortogonal de P en la varietat (columna), `distancia`=distància de P a la varietat.

Comandes notables: La comanda `null(A)` retorna una matriu que té una base del nucli de A *per columnes*.

Si teniu dues matrius A, C amb el mateix nombre de columnes, i les voleu posar una sobre l'altra en una matriu única, ho podeu fer amb una comanda `Mgran=[A;C];`.

Verificació: La projecció ortogonal del punt $(8, -1, 4)$ sobre la recta $\{3x - y + 4z = 2, x + 3y + 5z = 6\}$ és $(2.76667, 2.61373, -0.92157)$, que es troba a distància 8.0417. La projecció ortogonal d'un punt que ja és de la varietat és ell mateix.

Pràctica 2: Nivell de clor (amb marge d'error)

Termini d'entrega: 2010/3/5

Una piscina té la concentració de clor mantinguda automàticament per un mecanisme que pot afegir una quantitat regulable de clor cada dia, però que no disposa d'anàlitzador de

clor. Aquest mecanisme funciona tot sol sèries llargues de dies, i de tant en tant és corregit per un tècnic que analitza l'aigua i ajusta el nivell de clor.

Per a controlar el nivell de clor a la piscina entre anàlisi i anàlisi, es modela segons l'equació

$$u_{k+1} = \rho_k u_k + c_{k+1}$$

on u_k és el nivell de clor (g/l) a l'inici del dia k -èssim de la sèrie, u_{k+1} el de l'inici del dia $(k + 1)$ -èssim, ρ_k la fracció que es conserva del dia anterior (que s'estima a partir de la temperatura de l'aigua, que regula l'evaporació de clor), i c_{k+1} és el clor que s'afegeix a l'inici del dia $k + 1$ (també en g/l).

Heu de fer una funció de MATLAB que rebi com a dades d'entrada la concentració inicial de clor u_1 , un vector amb les taxes diàries de conservació del clor $\rho = (\rho_1, \dots, \rho_{n-1})$ per un període de n dies (n qualsevol), i un altre vector $c = (c_2, \dots, c_n)$ amb les quantitats de clor afegides durant aquest període. La vostra funció

- plantejarà el sistema d'equacions lineals format per l'equació $u_1 =$ valor donat per l'usuari i les equacions de u_{k+1} per $k = 1, \dots, n - 1$,
- resoldrà aquest sistema,
- i calcularà el marge d'error absolut de la solució en norma sub-infinit.

Per a calcular el marge d'error en la solució, apliqueu la fórmula de l'error absolut en la resolució d'un sistema lineal $Ax = b$ amb error en A, b , de la secció 2.2 dels apunts. Useu que coneixem les quantitats de clor afegides c_k amb un marge d'error absolut de 0.1 i les taxes de conservació del clor ρ_k amb un marge d'error absolut de 0.05.

Queda prohibit resoldre el problema calculant les concentracions de clor u_2, u_3, \dots una a una amb un bucle `for` o similar. Resoleu-lo formant el sistema lineal, perquè el marge d'error del càlcul és tan important com el resultat en sí.

Nom del fitxer: `conclor.m`

Capçalera de la funció: `function [u,marge]=conclor(u1,r,c)`

INPUT: `u1`=concentració de clor a l'inici del dia 1, `r`=taxes de conservació del clor del dia 1 al $n - 1$ (fila), `c`=afegiments de clor del dia 2 al dia n (fila).

OUTPUT: `u`=concentracions de clor diàries de l'inici del dia 1 a l'inici del dia n (fila), `marge`=marge d'error màxim en cada component de `u`.

Comandes notables: La comanda `zeros` crea una matriu plena de zeros de la mida que se li demani. La comanda `diag` serveix per insertar un vector de coeficients en diagonal en una matriu, a la diagonal central o a una de paral.lela.

La comanda `norm(M,inf)` calcula la norma sub-infinit d'una matriu o vector.

Verificació: Si la concentració inicial de clor és 1.2 g/l, dels dies 2 a 7 afegim 0.1 g/l, i la taxa de conservació del clor és del 96% els dies 1,2,5,6, del 93% el dia 3, del 94% el dia 4, les concentracions de clor del dia 1 (la inicial ajustada pel tècnic) al dia 7 són

$$u = (1.2000 \quad 1.2520 \quad 1.3019 \quad 1.3108 \quad 1.3321 \quad 1.3789 \quad 1.4237)$$

i el marge d'error absolut de la solució en norma sub-infinit és de 2.8173 g/l (segons com plantegeu el sistema d'equacions lineals aquest marge d'error pot variar fins en un 25%).

Una altra verificació parcial és que la solució que obteniu calculant els valors u_2, u_3, \dots un a un, amb l'equació $u_{k+1} = \rho_k u_k + c_{k+1}$, ha de coincidir amb la que obteniu resolent el sistema.

Pràctica 3: Interpolació amb corbes de Bézier

Termini d'entrega: 2010/3/12

L'objectiu d'aquesta pràctica és calcular i representar una corba \mathcal{C}^1 que approximi una taula de punts $(x_1, y_1), \dots, (x_n, y_n)$ observats.

Farem això usant *corbes de Bézier cúbiques*, perquè aquest mètode proporciona una corba que, si be no passa exactament per la majoria dels punts, té la primera derivada contínua i que varia de manera suau. Això fa més estable a un sistema (mecànic, químic ...) que es basi en seguir la corba.

L'algoritme per a formar una corba spline de Bézier cúbica que seguirem és el següent:

Partirem de la taula de punts $(x_1, y_1), \dots, (x_{2m}, y_{2m})$ que ha d'aproximar la corba, que compleixen $x_1 < x_2 < \dots < x_{2m}$.

Trobarem polinomis de grau 3 $p_1(x), p_2(x), \dots, p_{m-1}(x)$ de manera que cada corba $y = p_k(x)$ approximi als punts observats en l'interval $x \in [\frac{x_{2k-1}+x_{2k}}{2}, \frac{x_{2k+1}+x_{2k+2}}{2}]$ i en els punts de connexió entre dos intervals els polinomis de cada costat tinguin el mateix valor i derivada primera.

Per a aconseguir això trobarem els punts promig $Q_0 = (\frac{x_{2k-1}+x_{2k}}{2}, \frac{y_{2k-1}+y_{2k}}{2})$, $Q_f = (\frac{x_{2k+1}+x_{2k+2}}{2}, \frac{y_{2k+1}+y_{2k+2}}{2})$, i imposarem a la corba $y = p_k(x)$

- que passi pels dos punts Q_0, Q_f ,
- que la seva tangent en Q_0 tingui el pendent del vector de (x_{2k-1}, y_{2k-1}) a (x_{2k}, y_{2k}) ,
- i que la seva tangent en Q_f tingui el pendent del vector de (x_{2k+1}, y_{2k+1}) a (x_{2k+2}, y_{2k+2}) .

Aquestes 4 condicions es tradueixen en 4 equacions lineals sobre els coeficients del polinomi $p_k(x) = ax^3 + bx^2 + cx + d$, i com veurem a teoria el sistema que formen sempre és compatible determinat. Enlloc de fer anar una taula de polinomis de Bézier com a l'algoritme original, simplement resoldrem el sistema lineal amb Matlab.

Quan ja tinguem calculats els polinomis $p_1(x), \dots, p_{m-1}(x)$, farem que Matlab dibuixi en un gràfic únic:

- els punts $(x_1, y_1), \dots, (x_{2m}, y_{2m})$, indicats amb un cercle cadascun,
- i les gràfiques de totes les corbes $y = p_k(x)$, $x \in [\frac{x_{2k-1}+x_{2k}}{2}, \frac{x_{2k+1}+x_{2k+2}}{2}]$, per $k = 1, 2, \dots, m-1$,

Així comprovarem visualment que la corba és \mathcal{C}^1 i quina mena d'aproximació ens donen els splines de Bézier cúbics.

Nom del fitxer: beziercub.m

Capçalera de la funció: function p=beziercub(x,y)

INPUT: \mathbf{x} =vector fila amb els valors de x observats (longitud parella), \mathbf{y} =vector fila amb els valors de y observats (longitud parella).

OUTPUT: p =matriu amb 4 columnes que te per fila k -èssima els coeficients del polinomi $p_k(x) = ax^3 + bx^2 + cx + d$ del spline de Bézier cúbic explicat més amunt.

Comandes notables: Si voleu guardar una de cada dos components d'un vector en vector apart podeu fer-ho amb una comanda estil `parells=v(2:2:2*m-2)`.

Per a superposar dibuixos successius en el mateix gràfic de Matlab heu d'executar primer la comanda `hold on`, despres totes les comandes de dibuix que vulgueu, i al final fer `hold off` per a que l'usuari no hi escrigui mes coses a sobre.

Per a dibuixar una successió de punts al pla assenyalant-los amb cercles enlloc d'unir-los per una corba, podeu fer `plot(x,y,'o')`.

Opcionalment: en tots dos casos podeu fer que la línea sigui mes gruixuda incloent una directiva `'LineWidth'` dins de la comanda `plot`, o dibuixar de color diferent incloent una opció de color, per exemple `plot(x,y,'or')`; a mes de dibuixar una 'o' en cada punt les fa de color vermell.

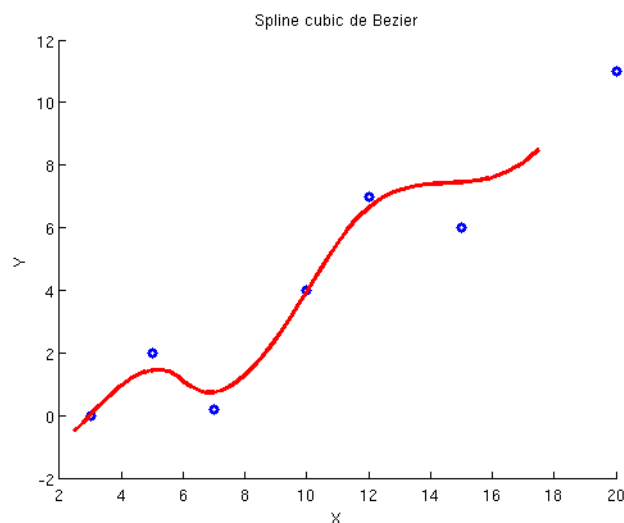
Si teniu un polinomi guardat com a vector fila de Matlab (en grau decreixent), i voleu evaluar-lo pels x d'una llista de valors xp podeu fer-ho amb `yp=polyval(polinomi,xp)`.

Verificació:

Aquest algoritme, amb dades inicials $x=[2,3,5,7,10,12,15,20]$; i $y=[-1,0,2,0.2,4,7,6,11]$; dóna matriu de polinomis

$$p = \begin{pmatrix} -0.0665 & 0.5761 & -0.6341 & -1.4767 \\ -0.0464 & 1.4232 & -12.9672 & 37.6904 \\ 0.0373 & -1.6340 & 23.9005 & -109.3641 \end{pmatrix}$$

i ha de dibuixar en pantalla els punts i corba de la figura (Matlab 7.5 es deixa el primer punt sense dibuixar, no us estranyeu si la vostra versió te alguna pega similar).



Pràctica 4: Filtratge i detecció de màxims en un senyal

Termini d'entrega: 2010/3/19

Mesurem una funció $y(x)$ en una malla de valors de x equiespaiada $x_1, x_2 = x_1 + dx, \dots, x_f = x_1 + (f-1)dx$. Aquesta mesura ve amb un soroll de fons, que suposem que és una oscil·lació periòdica de promig zero que es superposa al valor correcte.

Volem trobar els màxims locals de la funció $y(x)$. Per atenuar el soroll de fons, primer de tot farem un *promitjat dels valors de y*: si esperem que l'error es repeteixi cada m mesures amb promig 0 canviarem el vector de mesures $\{y_1, y_2, \dots, y_f\}$ per un vector de promitjos $\{\frac{y_1 + \dots + y_m}{m}, \frac{y_2 + \dots + y_{m+1}}{m}, \dots, \frac{y_{f-m+1} + \dots + y_f}{m}\}$. Aplicarem el mateix promitjat als valors de x per a assignar la coordenada x a cada promig de y .

Un cop filtrat el senyal volem trobar els màxims locals descartant les fluctuacions molt petites. Per a fer això, fixarem un *ample de la finestra w* i declararem màxims locals els valors de y filtrada que siguin més grans estrictament que els w valors anteriors i els w valors posteriors de la successió filtrada.

La funció de Matlab que heu de fer aquesta setmana rebrà com a dades els valors de $y(x)$ mesurats, l'interval $[x_0, x_f]$ en que s'han mesurat equiespaiadament, les amplades m del promitjat i w de la finestra per admetre màxims. La funció aplicarà el procediment descrit per a filtrar el senyal i identificar màxims locals, dibuixarà per pantalla

- el senyal $y(x)$ sense filtrar, amb línia fina,
- el senyal y_{filtrat} , amb línia més gruixuda,
- els màxims locals identificats, amb un cercle 'o' a cadascun.

Finalment, la funció retornarà a l'usuari dos vectors amb les coordenades x, y dels màxims locals trobats. Si no se n'ha trobat cap haurà de retornar vectors buïds.

Nom del fitxer: maxfiltrat.m

Capçalera de la funció: function [xmax,ymax]=maxfiltrat(y,x1,xf,m,w)

INPUT: y=vector fila amb els valors de y mesurats, x1,xf=extrems de l'interval de valors de x que cobreix la mesura, m=longitud del promitjat que fem al senyal, w=amplada per cada banda de la finestra per a identificar màxims.

OUTPUT: xmax,ymax=vectors fila amb la llista de coordenades x, y respectivament dels màxims locals trobats a la sèrie filtrada (vectors buïds si no n'ha aparegut cap).

Comandes notables: Per a crear un vector buïd heu de fer `xmax=[];`, després podeu afegir-li components o deixar-lo buïd.

Si v és un vector amb m uns, la comanda `conv(y,v/m)` produeix un vector que si li treieu els $m-1$ coeficients inicials i finals és el vector de promitjos buscat.

Verificació: Si feu `x=linspace(0,100,201);`,

`y=x.^ 1.25.*sin(x/3)+9.25*sin(x)+2.8*cos(x).*sin(6*x);` (els dos darrers sumands representen sorolls de fons), i crideu a `maxfiltrat` amb aquesta y , `x1=0`, `xf=100`, `m=12`, `w=6` heu d'obtenir com a resultats

- `xmax = 6.2500 23.7500 42.7500 61.2500 80.2500`
- `ymax = 6.4127 43.5044 90.7248 143.9579 201.5445`

i una figura com:

