Počítačové výpočty v řešení procesního projektu

použití Excelu, Matlabu a MAPLE

Jan Haidl

VŠCHT Praha

2019

Klasifikace typických úloh PP

- interpolace (extrapolace) dat - časový vývoj veličin, rovnovážná data, vyhodnocení kinetických parametrů
- řešení soustav lineárních rovnic - bilanční rovnice, numerické aproximace derivací v diferenciálních rovnicích
- řešení soustav nelineárních rovnic - výpočty patrových kolon, potrubních linek aj.
- integrace obyčejných diferenciálních rovnic - výměníky tepla, plněné kolony, trubkové či vsádkové reaktory
- integrace parciálních diferenciálních rovnic - neustálené vedení tepla či hmoty, dynamika chemických reaktorů

Interpolace dat

Proložení diskrétních dat křivkou:

- libovolnou hladkou (polynom, mocninná funkce, spline) - interpolace diskrétních dat, popisuje data rovnicemi,usnadňuje výpočet derivací a integrálů
- odvozenou z fyzikální představy o datech - fyzikálně konzistentní interpolace, lze opatrně extrapolovat (kubické rovnice, Clausiova-Clapeyronova rovnice, Arrheniova rovnice atp.)

Lineární regrese

proložení dat přímkou, polynomem:

```
> xData:=[0, 0.1, 0.2, ...1]:

> yData:=[0, 0.15, 0.27, ...1]:
```

- možné použít balík `with(Statistics)` a funkci `LinearFit` - vyžaduje data jako vektory (či matice)

  ```
  > y:=LinearFit([x,x^2,x^3],Vector(xData),Vector(yData),x)
  ```

  $$y := 0.11x + 0.023x^2 - 0.0012x^3$$

- případně lze použít balík `with(CurveFitting)` a funkci `LeastSquares`

  ```
  > y:=LeastSquares(xData,yData,x,curve=a*x+b*x^2+c*x^3)
  ```

  $$y := 0.11x + 0.023x^2 - 0.0012x^3$$

V MATLABu řešení přeurčené soustavy rovnic

$$\mathbf{X}\vec{p} = \vec{y}$$

```
» xData=[0, 0.1, 0.2, ...1];

» yData=[0, 0.15, 0.27, ...1];

» X=[xData; xData.^2; xData.^3]';

» p=X\yData';

p =
   0.11 0.023 -0.0012
```

Spline

interpolace dat hladkou křivkou, která projde všemi body (oproti regresi dat)

- křivka je množina polynomů konstruovaných mezi zadanými body
- spline stupně 1 je lomená čára mezi body - lineární interpolace => v hraničních bodech nemá první (ani vyšší) derivaci
- běžně je používána spline stupně 3: kubická - v uzlových bodech má spojité první derivace

V Maple balík `with(CurveFitting)` a funkce `Spline`

```
> y:=Spline(xData,yData,x);
```

$$y := \begin{bmatrix} 0.012x^3 - 0.01x^2 + 1.01x & x < 0.1 \\ 0.31(x-0.1)^3 - 0.12(x-0.1)^2 + 0.21x & x < 0.2 \\ \vdots & \end{bmatrix}$$

V Matlabu funkce **spline**: spočítá a uloží data do objektu, vyhodnocení v bodě poskytne funkce **ppval**

```
» S=spline(xData,yData);

» y=ppval(S,0.2)

y =
   0.27
```

Nelineární regrese

proložení dat obecnou křivkou metodou nejmenších čtverců
V Maple balík `with(Statistics)` a funkce `NonlinearFit`

```
> y:=NonlinearFit(a*x+b*x^c,xData,yData,x);
```

$$y := 1.12x + 3.22x^{0.177}$$

V Matlabu funkce **lsqcurvefit**

```
» F = @(p,x)p(1)*x+p(2)*x.^p(3); %definice křivky

» p0 = [1,1,1]; % počáteční odhad parametrů

» p = lsqcurvefit(F,p0,xData,yData)

p =
   1.12 3.22 0.177
```

Řešení soustav lineárních rovnic

- obvykle řešíme soustavy $\mathbf{A}\vec{x} = \vec{y}$ se čtvercovou maticí
- v Maple řešíme buď soustavu rovnic funkcí `solve`,

```
> eq1:=x+y=3:
> eq2:=x-y=-1:
> solve({eq1,eq1});
```

$$\{x = 1, y = 2\}$$

- nebo maticově zadanou soustavu funkcí `LinearSolve` z balíku with(LinearAlgebra)

```
> A:=Matrix([[1,1],[1,-1]]):
> y:=Vector([3,-1]):
> x:=LinearSolve(A,y);
```

$$x := \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

- v Matlabu mocný operátor \

```
» A=[1 1;1 -1];
» y=[3 -1]';
» x=A\y
x =
    1
    2
```

- nikdy neřešíme pomocí inverzní matice
- pro opakované řešení soustav s konstantní maticí **A** a proměnlivou pravou stranou $\vec{y}$ je výhodné předřešit problém pomocí LUP rozkladu, či QR rozkladu

```
> P,L,U:=LUDecomposition(A):
```

```
» [L,U,P]=lu(A);
```

```
» b=L\(P*y)
b =
   3
   -4
```

```
» x=U\b
x =
   1
   2
```

- pro velké problémy - matice 100x100 a větší - můžete výrazně zkrátit dobu výpočtu určením vhodného pořadí operací:

```
» (A*A*A)*y; % dvojnásobný součin matic a poté součin matice a vektoru
```

$2n^3 + n^2$ operací

```
» A*(A*(A*y)); % trojnásobný součin vektoru s maticí
```

$3n^2$ operací

Řešení nelineárních rovnic a soustav

- vždy iterativní postup
  - rovnice jedné proměnné - jednoduché řešení metodou půlení intervalu
  - v Matlabu funkce **fzero**

```
» y = @(x)exp(x)-0.5; %inline definice funkce jedné proměnné
» fzero(y,0) %počáteční odhad řešení, x0=0
ans =
   -0.6934
```

  - soustavy rovnic - Newtonova metoda - v Maple i Matlabu funkce `fsolve`
  - v Maple lze (bývá potřeba) omezit interval pro jednotlivé proměnné

```
> fsolve({eq1,eq2},{x,y},{x=-5..5,y=-3..3});
```

$$\{x = 1, y = 2\}$$

  - Matlab vyžaduje definici funkce v anulovaném tvaru a vhodný nástřel

```
function [y] = myfun(x)
   y(1)=x(1)+x(2)-3
   y(2)=x(1)-x(2)+1
end
```

```
» fsolve(@(x)myfun(x),[0 0])
ans =
   1.0000
   2.0000
```

- pro úspěšné řešení je třeba dohlédnout na správný počet a typ řešených rovnic - nenechávejte řešit duplicitní rovnice!
- metoda fsolve v Maple může vyžadovat rozumné omezení limitů na proměnné
- řešitelnosti soustav můžete značně pomoci předřešením

  ```
  > eq1:=x^2+y=0: eq2:=x+y=0: fsolve({eq1,eq2});
  ```

  řeší dvě rovnice pro dvě proměnné

  $$\{x = 1.0000, y = -1.0000\}$$

  ```
  > y:=-x^2: eq2:=x+y=0: fsolve({eq2});
  ```

  řeší jednu rovnici pro jednu proměnnou

  $$\{x = 1.0000\}$$
- **vyhněte se přiřazovacím rovnicím!**, využívejte sekvenční přístup k řešení rovnic

  ```
  > eq1:=y=-x^2:
  ```
- velmi silný nástroj - **Řešitel** - pro řešení nelineárních soustav a optimalizaci nabízí MS Excel
- úlohy jsou zde vždy definovány jako optimalizační úlohy, tj. úlohy minimalizující účelovou funkci změnou až 100 parametrů
- na parametry je možno klást omezující podmínky (defaultně je zapnuta podmínka **nezápornosti parametrů**)
- umožňuje definovat tzv. Multistart pro obejití lokálního minima
- pro zajištění nalezení optima je vhodné pustit nástroj opakovaně - restart může překonat lokální minimum

Soustavy diferenciálních rovnic - ODR

- soustavu rovnic libovolného řádu lze převést na soustavu ODR 1. řádu
- řešíme soustavu rovnic
$$\vec{y'} = \vec{f}(\vec{x}, \vec{y}, \vec{y'})$$
- soustavy lineárních diferenciálních rovnic umíme řešit analyticky
  - hledáme ve tvaru
$$y(\vec{x}) = \sum_{i=1}^{N} C(x) \mathrm{e}^{\lambda_i x}$$
  - v Maple pomocí funkce `dsolve` z balíku `with(DETools)`
  - diferenciální rovnice zadáváme pomocí funkce `diff`, nebo operátoru `D`

```
> ODEs:=D(x)(t)=x(t)-y(t), D(y)(t)=x(t)+y(t):
> ODEs:=diff(x(t),t)=x(t)-y(t), diff(y(t),t)=x(t)+y(t);
```
$$ODEs := \frac{\mathrm{d}}{\mathrm{d}t}x(t) = x(t) - y(t), \frac{\mathrm{d}}{\mathrm{d}t}y(t) = x(t) + y(t)$$
```
> dsolve(ODEs)
```
$$\{x(t) = \mathrm{e}^{t}(\_C2\cos(t) + \_C1\sin(t)), y(t) = -\mathrm{e}^{t}(\cos(t)\_C1 - \sin(t)\_C2)\}$$
  - lze hledat i partikulární řešení

```
> ICS:=x(0)=1,y(1)=0:
> dsolve(ODEs,ICS);
```
$$\left\{x(t) = \mathrm{e}^{t}\left(\cos(t) + \frac{\sin(1)\sin(t)}{\cos(1)}\right), y(t) = -\mathrm{e}^{t}\left(\frac{\cos(t)\sin(1)}{\cos(1)} - \sin(t)\right)\right\}$$

- v Matlabu lze podobně využít symbolické výpočty pomocí funkcí **syms** a **dsolve**

```
» syms x(t) y(t);
» odes=[diff(x)=x-y); diff(y)=x+y];
» conds=[x(0)==1,y(1)==0];
» [xS,yS]=dsolve(odes,conds);
xS(t) =
   exp(t)*cos(t) + (cos(t - 1)*exp(t))/(2*cos(1)) - (cos(t +
1)*exp(t))/(2*cos(1))

yS(t) =
   exp(t)*sin(t) + (sin(t - 1)*exp(t))/(2*cos(1)) - (sin(t +
1)*exp(t))/(2*cos(1))
```

- soustavy nelineárních rovnic řešíme obvykle numericky
  - pro řešení doporučováno použít integraci metodou Runge-Kutta 4. řádu
  - v Maple řešeno opět funkcí `dsolve` s možností `numeric`
  - Maple vrací proceduru, která spočítá požadovanou hodnotu funkce až v okamžiku zavolání

```
> sol:=dsolve(ODEs,ICS,numeric);
```

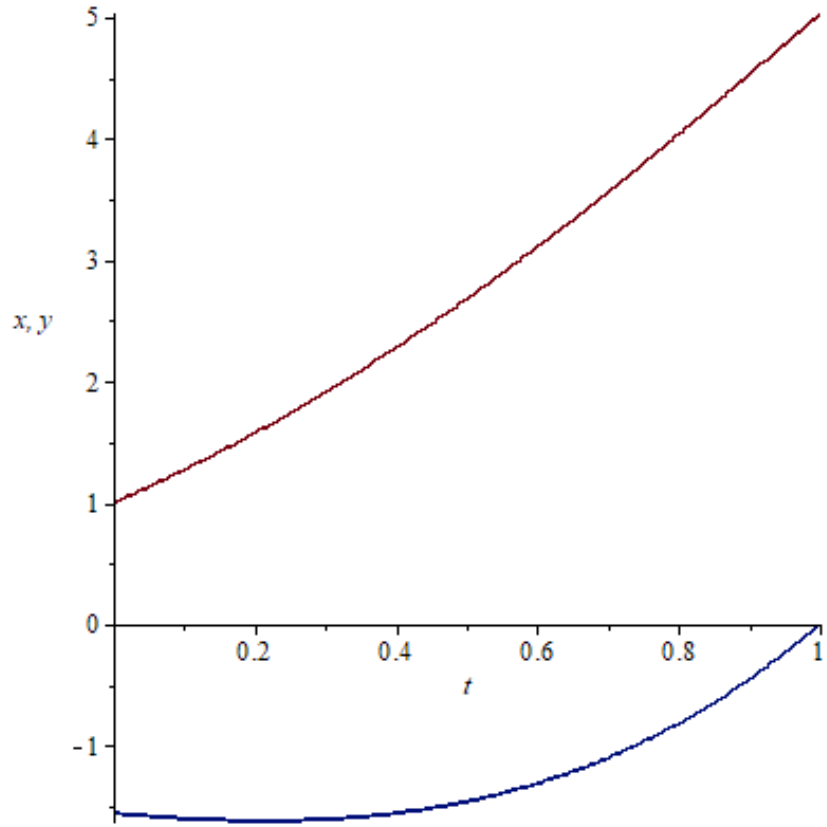
$$sol := \mathbf{proc}(x\_bvp)...\mathbf{endproc}$$

```
> sol(0.1);
```

$$[t = 0.1, x(t) = 1.27148310772961848, y(t) = -1.60226986648556902]$$

- řešení je možné vykreslit pomocí funkce `odeplot` z balíku `with(plots)`

```
> odeplot(sol,[[t,x(t)],[t,y(t)]],t=0..1);
```



- případně animovat přidáním parametru `frames`

```
> odeplot(sol,[[t,x(t)],[t,y(t)]],t=0..1,frames=40);
```

- v Matlabu je metoda Runge-Kutta skrytá ve funkci **ode45**
- soustavu rovnic je třeba definovat jako *sloupcovou* vektorovou funkci $\vec{y}' = \vec{f}(t, y)$

```
function dy = odes(t,y)
   dy=zeros(2,1);
   dy(1)=y(1)-y(2);
   dy(2)=y(1)+y(2);
end
```

- integrace vrací vektor času a matici řešení
- ode45 řeší pouze počáteční úlohu
- řešení úloh s okrajovými podmínkami je možné metodou střelby: kombinace **fsolve** a **ode45**
- syntaxe ode45(@funkce, [tMin;tMax],y0)

```
» [t,Y]=ode45(@odes,[0;1],[1;-1.5])
```

- řešení je možné vykreslit pomocí **plot**

```
» plot(T,Y)
```

- metoda si sama volí délku integračního kroku - vektor T je tedy vracen dle potřeb metody
- chcete-li vyčíslit řešení ve specifikovaných bodech, je možné místo intervalu [tMin;tMax] vypsat požadované časy [tMin;t1;t2;t3;tMax] či ekvidistantní interval [tMin:deltaT:tMax]

Soustavy diferenciálních rovnic - PDR

$$A\frac{\partial^2 u}{\partial x^2} + B\frac{\partial^2 u}{\partial x \partial y} + C\frac{\partial^2 u}{\partial y^2} + D\frac{\partial u}{\partial x} + E\frac{\partial u}{\partial y} = f(x, y)$$

- analytické řešení je známé jen pro velmi omezené spektrum lineárních PDR (např. Fourierova metoda - viz MCHI)
- v praxi obvykle řešeny numericky
- metody se liší dle typu rovnice
  - eliptické: $B^2 - 4AC < 0$ - Laplaceova rovnice, ustálené rozložení teploty
  - parabolické: $B^2 - 4AC = 0$ - neustálené vedení tepla, NS rovnice v laminární oblasti
  - hyperbolické: $B^2 - 4AC > 0$ - vlnové rovnice, NS rovnice mimo laminární oblast
- v Maple funkce z balíku `PDETools`
- numerické metody řešení jsou založeny na:
  - diskretizaci domény - síťování
  - diskretizaci rovnic - aproximaci jednotlivých členů lineárními funkcemi
  - řešením soustav LAG
- poměrně univerzální metody:
  - metoda konečných prvků - FEM
  - metoda konečných objemů - FVM

- při řešení neustálených dějů nutné kontrolovat délku časového kroku, při použití příliš velkého kroku řešení obvykle diverguje
- dobré kritérium sledování délky kroku - Courantovo číslo

$$Co = \Delta t \sum \frac{u_i}{\Delta x_i} < 1(0.5)$$

  u_i - rychlost šíření informace ve směru i, x_i - rozměr buňky sítě ve směru i
- v Matlabu pro řešní lineárních problémů v 1D a 2D je dostupný grafický nástroj: **pdetool** - poučívá trojúhelníkovou síť a FEM
- pro 3D problémy a neustálené problémy je možné použít Comsol - obvykle FEM, umí i FVM
- pro ostatní problémy nutné použít sofis