



UPPSALA
UNIVERSITET

Problemlösning 3 - Kurvanpassning

Daniel Elfverson
`daniel.elfverson@it.uu.se`

Division of Scientific Computing
Uppsala University
Sweden

Gillespies algoritim for stokastiska simuleringar

Algorithm 1 Gillespies algoritim

- 1: Initialvilkor: sätt tillståndsvektorn $x = x_0$, starttid, $t = t_0$ och stopp tid T
 - 2: **while** $t < T$ **do**
 - 3: Sampla τ , när nästa reaktion inträffar
 - 4: Hitta vilken reaktion som inträffar
 - 5: Uppdatera tillståndsvektorn och tiden
 - 6: **end while**
-

Svårigheter

- 1 Okänt (varierande) antal tidssteg
- 2 Hitta vilken reaktion som inträffar
- 3 Plotta lösningen

Okänt (varierande) antal tidssteg

- Gissa hur många tidssteg och allokeras minne
- Gissa hur många tidssteg och allokeras minne, allokeras nytt när minnet är slut
- Bestäm vid vilka tidpunkter du vill spara lösningen och allokeras minne

Gissa hur många tidssteg och allokerar minne

```
1 N = 250000          ( Gissning )  
2 t = zeros (N,1) ;  
3 A = zeros (N,1) ;  
4 R = zeros (N,1) ;
```

Allokera nytt när minnet är slut

```
1 N = 250000;
2 NExpand = 250000;
3 tPlot = zeros(1,N);
4 APlot = zeros(1,N);
5 RPlot = zeros(1,N);
6
7 t = 0
8 tFinal = 200;
9 n = 0;
10 while t < tFinal
11     ...
12     ...
13     n = n + 1;
14     if n > N
15         tPlot = [tPlot zeros(1,NExpand)];
16         ...
17         N = N + NExpand;
18     end
19 end
```

Lösning sparas vid bestämda tidpunkter

```
1 N = 100;
2 tPlot = zeros(1,N);
3 APlot = zeros(1,N);
4 RPlot = zeros(1,N);
5
6 t = 0
7 tFinal = 200;
8 deltaT = (tFinal-t)/N;
9 n = 1;
10 while t < tFinal
11     ...
12     ...
13
14     if n*deltaT <= t
15         tPlot(n) = t;
16         ...
17         n = n + 1;
18     end
19 end
```

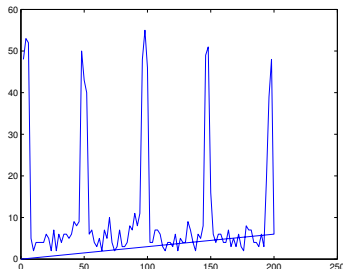
Hitta vilken reaktion som inträffar

- Löstes på många olika sätt
- Tips på hur den kan lösas med 'cumsum' och 'find' (bra att träna på 'find' till M3)

```
1 ...  
2 u2 = rand();  
3 w = prop_vilar(x);  
4 a0 = sum(w);  
5 ...  
6 F = cumsum(w);  
7 r = find( u2*a0 <= F,1);  
8 ...
```

Plotta lösningen

- Mystiskt sträck som gick genom plotten (unkown annoying object UAO)



```
1 t = [0.5, 1, ..., 200, 0, 0]
2 u = [ 48, 50, ..., 1800, 0, 0]
3 plot(t, u)
```


Miniprojekt 3 - Kurvanpassning

- Uppgift - Hitta ett säsongsberoende av Dikväveoxid, N_2O (lust gas), i atmosfären.
- Växthusgas som är 200 gånger starkare än koldioxid.
- Baseras på *Liao T., Camp C. D., Yung. Y. L.: The seasonal cycle of N_2O* (se kurshemsida)

Kurvanpassning - repetition

Olika sätt att approximera med polynom

- Newton interpolation (ett polynom som går genom mätpunkterna)
- Minsta kvadratanpassning (ett polynom som är ett typ av medelvärde)
- Spline (styckvisa polynom)

Minsta kvadratanpassning

- Används då man har en "stor" datamängd.
- Anpassa ett 2:a grads polynom, ansats $p_2(x) = a_0 + a_1x + a_2x^2$.

Där $x = [1, 2, 3, 4, 5]$ kan tex vara tiden för mätningarna och $y = [1, 2, 1, 2, 3]$ vara mätvärden, men får så ett överbestämt system

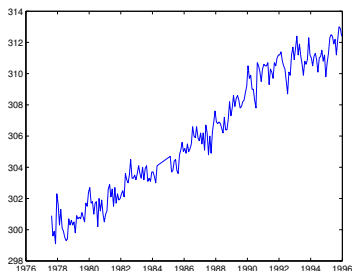
$$\begin{aligned} a_0 + a_1 \cdot 1 + a_2 \cdot 1^2 &= 1 \\ a_0 + a_1 \cdot 2 + a_2 \cdot 2^2 &= 2 \\ a_0 + a_1 \cdot 3 + a_2 \cdot 3^2 &= 1 \\ a_0 + a_1 \cdot 4 + a_2 \cdot 4^2 &= 2 \\ a_0 + a_1 \cdot 5 + a_2 \cdot 5^2 &= 3 \end{aligned} \Rightarrow \begin{pmatrix} 1 & 1 & 1^2 \\ 1 & 2 & 2^2 \\ 1 & 3 & 3^2 \\ 1 & 4 & 4^2 \\ 1 & 5 & 5^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 1 \\ 2 \\ 3 \end{pmatrix}$$

- Löses med hjälp av normalekvationen $A^T A a = b$.
- Normalekvationen hittar v så att $\|b - Av\|_{L^2}$ minimeras.

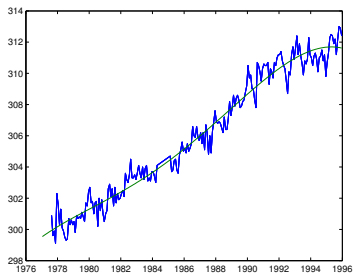
Löses i matlab med hjälp av

```
1 p = polyfit(x, y, 2)
```

Avtrendifiering



- Uppgift - Hitta ett säsongsberoende av Dikväveoxid, N_2O (lust gas), i atmosfären.
- Hitta den årliga trenden (genom minsta kvadratanpassning) och dra bort den från datan.



- Data (blå) med årlig trend (grönt).

Programmerings tips

- `load`: För txt filer så måste all onumerisk data bort, då kan `load('namn.txt')` användas.

A bort önskade data

```
1 x = [1      2 3  4    5  6];  
2 y = [100 120 0 140  0 135];  
3  
4 i = find(y ~= 0);  
5  
6 xnz = x(i);  
7 ynz = y(i);
```

- Vad är xnz och ynz?

Index- och logiska vektorer

Indexvektorer

Standard vektorn som nu är vana att jobba med, kan se ut som

$$x = [1 \ 2 \ 0 \ 1 \ 2 \ 0 \ 9 \ 1 \ 0];$$

och $x(4) = 1$.

Logiskavektorer

Element som är 1 eller 0. Operationen

$y = (x \sim= 0)$ ger en logisk vektor om är

$$y = [1 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0];$$

Bra för högprestanda beräkningar

Deadline: 7 juni (samma datum som tenta, så det rekommenderas att lämna in tidigare).