Miniprojekt 3

Beräkningsvetenskap II

Per Engström & Nathalie Proos Vedin

11 mars 2015

Innehåll

1. Inledning

Dikväveoxid, N_2O, är en mycket potent växthusgas. Den är som molekyl per molekyl 200 gånger så stark som koldioxid. Det är därför av största intresse att kartlägga dess beteende i atmosfären. I denna rapport kommer vi undersöka om mängden N_2O i atmosfären är säsongsberoende.

Vi använder data från *National Oceanic and Atmospheric Administration – Climate Monitoring and Diagnostics Laboratory (NOAA-CMDL) Global Cooperative Air Sampling Network* från 1977 till 2000. Datan avtrendifieras med en polynomiell minsta-kvadratanpassning och undersöks sedan månadsvis.

2. Matematisk modell

Centralt i vår metod är en minsta-kvadratanpassning. Det är ett verktyg för att anpassa en funktion $f(x)$ med okända parametrar till mätdata (x_i, y_i) på *bästa sätt*. I detta fall betyder *bästa sätt* att kvadratsumman av residualerna r minimeras. Residualerna r_i definieras som skillnaden mellan datan y_i och det framtagna funktionsvärdet $f(x_i)$ enligt

$$r_i = y_i - f(x_i) \tag{1}$$

I vårt fall vill vi anpassa ett polynom av grad n till datan. Funktionen $f(x)$ har då formen

$$f(x) = a_0 + a_1 x + \dots a_n x^n \tag{2}$$

och vi har därför $n+1$ parametrar att bestämma. Kvadratsumman av residualerna kan då skrivas som

$$F(\mathbf{a}) = \sum_i (y_i - f(x_i))^2 = \sum_i (y_i - (a_0 + a_1 x_i + \cdots + a_n x_i^n)^2 \tag{3}$$

Nödvändigt villkor för minimum är $\partial F/\partial a_j = 0$. Detta ger oss $n+1$ ekvationer för $n+1$ okända.

$$\frac{\partial F}{\partial a_0} = \sum_i \Big[2(y_i - (a_0 + a_1 x_i + \cdots + a_n x_i^n))(-1)\Big] = 0 \tag{4}$$

$$\frac{\partial F}{\partial a_1} = \sum_i \Big[2(y_i - (a_0 + a_1 x_i + \cdots + a_n x_i^n))(-x_i)\Big] = 0 \tag{5}$$

$$\vdots \tag{6}$$

$$\frac{\partial F}{\partial a_n} = \sum_i \Big[2(y_i - (a_0 + a_1 x_i + \cdots + a_n x_i^n))(-x_i^n)\Big] = 0 \tag{7}$$

Vi kan skriva om dem som

$$a_0 \sum_i 1 + a_1 \sum_i x_i + \cdots + a_n \sum x_i^n = \sum_i y_i \tag{8}$$

$$a_0 \sum_i x_i + a_1 \sum_i x_i^2 + \cdots + a_n \sum x_i^{n+1} = \sum_i x_i y_i \tag{9}$$

$$\vdots \tag{10}$$

$$a_0 \sum_i x_i^n + a_1 \sum_i x_i^{n+1} + \cdots + a_n \sum x_i^{2n} = \sum_i x_i^n y_i \tag{11}$$

Eller på matrisform

$$\begin{bmatrix} \sum 1 & \dots & \sum x_i^n \\ \vdots & \ddots & \vdots \\ \sum x_i^n & \dots & \sum x_i^{2n} \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} \sum y_i \\ \vdots \\ \sum x_i^n y_i \end{bmatrix} \tag{12}$$

som är ett enkelt system och lätt att lösa.

2.1. Val av polynomgrad

Vi är fria att välja vilken grad av polynom vi vill, förutsatt att graden n uppfyller $n \leq m-1$ för antal datapunkter m. Vid likhet behövs ingen minsta-kvadratanpassning, utan lösningen är exakt.

Enligt [2] bör vi välja den grad som minimerar variansen

$$\frac{\mathrm{Sr}(n)}{m-n-1} \tag{13}$$

för kvadratsumman av residualerna $\mathrm{Sr}(n)$.

3. Numeriska metoder

Vi kan lösa matrisekvationen 12 direkt i MATLAB, men när polynomgraden växer kommer matrisen vi inverterar att bli mer och mer instabil. Detta kan motverkas genom att ortogonalisera den, vilket görs i MATLAB:s `polyfit`. Våra mätvärden är också täta, vilket försvårar minsta-kvadratanpassningen. Men även det kan motverkas genom att centrera datan innan anpassning, vilket görs automatiskt av `polyfit`.

Vår data behöver behandlas eftersom vissa datapunkter saknas. Dessa utmärks med att värdet är 0. Vi vill därför ta ut de nollskilda elementen med `nz = find(data ~= 0)` vilket ger deras index. Vi har då vektorer med x och y värden för de datapunkter

där $y \neq 0$. Då gör vi en minsta-kvadratanpassning genom `p = polyfit(x, y, M)` för polynom av grad `M`.

Vi kan då avtrendifiera datan genom att subtrahera trenden (det anpassade polynomet) från datan `detrend = y - polyval(p,x)`. Sedan lägger vi tillbaka de avsaknade mätvärdena på samma sätt som innan och tar medelvärdet av varje månad. Koden nedan itererar först över varje månad, och sedan var 12:e element i datan. Den summerar i `sum_months` om värdet är skiljt från 0. En räknare `numval` håller reda på antalet termer i summan. `N` är antalet ursprungliga datapunkter och `mx_det` är den avtrendifierade datan (med nollor återinförda).

```
for i=1:12
    for j=i:12:N
        if (mx_det(j) ~= 0)
            sum_months(i) = sum_months(i) + mx_det(j);
            numvals(i) = numvals(i) + 1;
        end
    end
end

months_avg = sum_months ./ numval;
```

3.1. Val av polynomgrad

Vi kan implementera ekvation 13 som

```
p = polyfit(x, y, N);
res = y - polyval(p, x);
sr = sum(res.^2);
var = sr/(M - N - 1);
```

för polynom av grad `N` med antal datapunkter `M`.

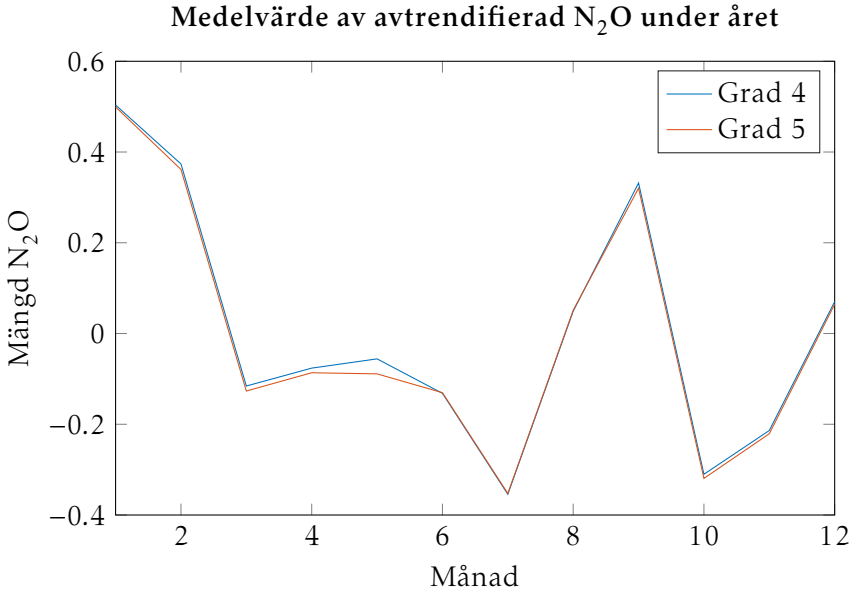
4. Indata

För att importera indata från filen `Old_GC_Globals_2001_prn.txt` skapades en funktion som returnerar varje kolumn som en vektor. Koden, som kan ses i bilagan, använder MATLAB-kommandot `textscan` för att tolka datan.

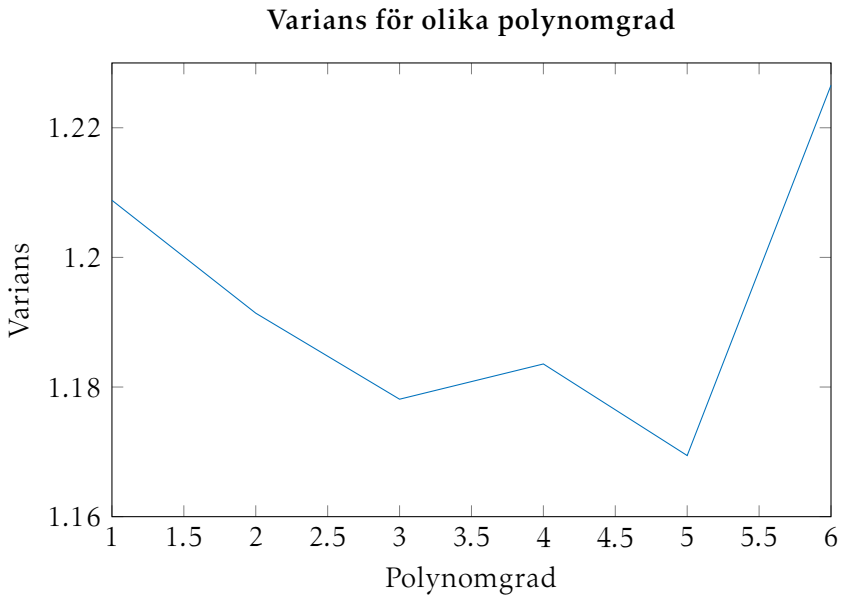
Eftersom vi båda är födda i den senare halvan av månaden så använder vi data från södra halvklotet (`sh_mx` i koden).

5. Resultat

I figur 1 ser vi resultatet av detrendifieringen och månadsvist medelvärde. Detrendifiereingen gjordes med ett polynom av grad 5 enligt resultat i figur 2 och grad 4 efter valet i [1].



Figur 1: Medelvärdet av varje månads nivå av N_2O beräknad från avtrendifierad data.



Figur 2: Variansen enligt ekvation 13.

6. Diskussion

Enligt [2] och figur 2 bör vi välja ett polynom av grad 5. Detta eftersom det minimerar variansen enligt ekvation 13 för gradtal upp till 6. I rapporten väljer de dock ett polynom av grad 4 till datan från norra halvklotet eftersom det liknar mycket polynomet av grad 5, och vi vill så långt vi kan välja ett lågt gradtal för att undvika överanpassning. Dock gjorde de ingen anpassning av datan från södra halvklotet eftersom felet blev för högt.

Vi jämför de båda graderna för södra halvklotet i figur 1, vilket avslöjar att skillnaden även här är liten, och antagligen inom felmarginalen.

När vi försökte anpassa datan med `polyfit` fick vi en varning om att polynomet var dåligt konditionerat. Detta beror antagligen på att datapunkterna är täta och svåra att skilja på. Detta åtgärdas genom att anropa `polyfit` på ett speciellt sätt:

```
[p, ~, mu] = polyfit(x, y, M);
```

Detta skiftar ock skalar om datan så att konditionen blir bättre. Tecknet ~ signalerar att vi inte behöver det andra elementet och `mu` innehåller information om skalningen och skiftningen. Haken är att vi även måste ändra hur vi anropar `polyval`:

```
f = polyval(p, x, [], mu);
```

Tredje argumentet är relaterat till elementet vi ignorerade med ~ ovan.

Referenser

[1] *The Seasona cycle of N_2O*, Ting Liao, Charles D. Camp and Yuk L. Yang, Geophysical Research Letters vol. 31

[2] *Finding the optimum polynomial order to use for regression*, Autar Kaw, 5 juli 2008, https://autarkaw.wordpress.com/2008/07/05/finding-the-optimum-polynomial-order-to-use-for-regression/

A. Huvudkod

```
clear;

%% Constants
M = 5; % Polynomial degree

%% Data import
[time,nh_mx,nh_sd,nhc,sh_mx,sh_sd,shc,gl_mx,gl_sd,glc,nhsh] ...
    = importfile('Old_GC_Globals_2001_prn.txt');

data = sh_mx;

%% Zero removal

nz = find(data ~= 0); % Returns indicies of nonzero value
x = time(nz); % Corresponding regressor
y = data(nz); % and response
N = length(x); % Number of data points

%% Fitting

[p, ~, mu] = polyfit(x, y, M);% Fit a polynomial of degree M
r_sq = sum((polyval(p,x,[],mu)-y).^2); % Calculate sum of residuals squared

%% Detrend

mx_det = zeros(1,length(data)); % Allocate detrended data
y_det = y - polyval(p, x, [], mu); % Subtract trend
mx_det(nz) = y_det; % Assign non-zero values

%% Monthly means

sum_months = zeros(1,12);
numval = zeros(1,12);

for i=1:12
    for j=i:12:N
        if (mx_det(j) ~= 0)
            sum_months(i) = sum_months(i) + mx_det(j);
            numval(i) = numval(i) + 1;
        end
    end
end

months_avg = sum_months ./ numval;

plot(1:12,months_avg);
title('Medelvärde av avtrendifierad N2O under året');
xlabel('Månad')
ylabel('Mängd N2O')
```

B. Dataimport

```
function [time,nh_mx,nh_sd,nhc,sh_mx,sh_sd,shc,gl_mx,gl_sd,glc,nhsh] =
    importfile(filename, startRow, endRow)
%IMPORTFILE Import numeric data from a text file as column vectors.
%   [TIME,NH_MX,NH_SD,NHC,SH_MX,SH_SD,SHC,GL_MX,GL_SD,GLC,NHSH] =
%   IMPORTFILE(FILENAME) Reads data from text file FILENAME for the default
%   selection.
%
%   [TIME,NH_MX,NH_SD,NHC,SH_MX,SH_SD,SHC,GL_MX,GL_SD,GLC,NHSH] =
%   IMPORTFILE(FILENAME, STARTROW, ENDROW) Reads data from rows STARTROW
%   through ENDROW of text file FILENAME.
%
% Example:
%   [time,nh_mx,nh_sd,nhc,sh_mx,sh_sd,shc,gl_mx,gl_sd,glc,nhsh] =
%   importfile('Old_GC_Globals_2001_prn.txt',2, 229);
%
%    See also TEXTSCAN.

% Auto-generated by MATLAB on 2015/03/03 18:20:48

%% Initialize variables.
if nargin<=2
    startRow = 2;
    endRow = inf;
end

%% Format string for each line of text:
%   column1: double (%f)
%       column2: double (%f)
%   column3: double (%f)
%       column4: double (%f)
%   column5: double (%f)
%       column6: double (%f)
%   column7: double (%f)
%       column8: double (%f)
%   column9: double (%f)
%       column10: double (%f)
%   column11: double (%f)
% For more information, see the TEXTSCAN documentation.
formatSpec = '%8f%6f%6f%4f%6f%6f%4f%6f%6f%4f%f%[^\n\r]';

%% Open the text file.
fileID = fopen(filename,'r');

%% Read columns of data according to format string.
% This call is based on the structure of the file used to generate this
% code. If an error occurs for a different file, try regenerating the code
% from the Import Tool.
dataArray = textscan(fileID, formatSpec, endRow(1)-startRow(1)+1, 'Delimiter'
    , '', 'WhiteSpace', '', 'HeaderLines', startRow(1)-1, 'ReturnOnError',
    false);
for block=2:length(startRow)
```

```
    frewind(fileID);
    dataArrayBlock = textscan(fileID, formatSpec, endRow(block)-startRow(
        block)+1, 'Delimiter', '', 'WhiteSpace', '', 'HeaderLines', startRow(
        block)-1, 'ReturnOnError', false);
    for col=1:length(dataArray)
        dataArray{col} = [dataArray{col};dataArrayBlock{col}];
    end
end

%% Close the text file.
fclose(fileID);

%% Post processing for unimportable data.
% No unimportable data rules were applied during the import, so no post
% processing code is included. To generate code which works for
% unimportable data, select unimportable cells in a file and regenerate the
% script.

%% Allocate imported array to column variable names
time = dataArray{:, 1};
nh_mx = dataArray{:, 2};
nh_sd = dataArray{:, 3};
nhc = dataArray{:, 4};
sh_mx = dataArray{:, 5};
sh_sd = dataArray{:, 6};
shc = dataArray{:, 7};
gl_mx = dataArray{:, 8};
gl_sd = dataArray{:, 9};
glc = dataArray{:, 10};
nhsh = dataArray{:, 11};
```

C. Val av polynomgrad

```
%% Data import

[time,nh_mx,nh_sd,nhc,sh_mx,sh_sd,shc,gl_mx,gl_sd,glc,nhsh] ...
    = importfile('Old_GC_Globals_2001_prn.txt');

data = sh_mx;

%% Zero removal

nz = find(data ~= 0); % Returns indicies of nonzero value
x = time(nz); % Corresponding regressor
y = data(nz); % and response

%% Loop

N = length(x);
var = zeros(1,6);
for M=1:6
```

```
    % Fitting
    p = polyfit(x, y, M);
    r_sq = sum((y-polyval(p,x)).^2); % Sum of square of residuals

    % Metric
    var(M) = r_sq / (N - M - 1);

end

plot(1:6, var);
title('Varians för olika polynomgrad');
xlabel('Polynomgrad');
ylabel('Varians');
```