Projekt: Fachwerksimulator in MATLAB

Vorlesung “Lineare Algebra und Numerische Mathematik”

19. Februar 2014

1 Problemspezifikation

In diesem Projekt sollen Sie ein MATLAB-Programm schreiben, das die Kräfte in einem statisch bestimmten planaren Fachwerk berechnet, wenn die Geometrie des Fachwerks, Art und Lage der Lager und die äusseren Kräfte gegeben sind. Statisch bestimmte Fachwerke wurden in Abschnitt 3.8.2 der Vorlesung behandelt.

Das Ergebnis des Projekts soll eine MATLAB-Funktion

```
function trusssimulator(jointpos,links,supports,...
                        suppplanes,forces)
```

sein, die folgende Funtionsargumente erwartet:

- `jointpos` ist eine $2 \times k$-Matrix, $k \in \mathbb{N}$ deren Spalten die Koordinaten der Gelenkpositionen enthalten.
- `links` ist eine $s \times 2$-Integermatrix $\in \{1, \ldots, k\}^{s,2}$, $s \in \mathbb{N}$, deren Zeilen Paare von Gelenkindices enthalten und dadurch die Lage der Stäbe beschreiben.
- `supports` ist ein Integervektor der Länge m, $m \in \mathbb{N}$, mit Einträgen aus $\{1, \ldots, k\}$, der die Nummern der Gelenke enthält, die gelagert sind.
- `suppplanes` ist eine $2 \times m$-Matrix $\in \mathbb{R}^{2,m}$, die zu jedem Lager entsprechend der Aufzählung in `supports` einen Normalenvektor angibt, der für Auflager die Ebene beschreibt, in der das Lager beweglich ist. Ist die j. Spalte von `suppplanes` gleich dem Nullvektor, dann bedeutet das, dass es sich beim j. Lager um ein festes Lager handelt.
- `forces` is eine $2 \times k$-Matrix, deren Spalten die Vektoren der externen Kräfte enthalten, die auf die einzeln Gelenke wirken.

Die Funktion soll folgende Bildschirmausgaben produzieren:

- Eine Grafik, die die Gelenke und Stäbe des Fachwerks visualisiert, so wie in Abbildung 1 demonstriert. Die Beschriftung von Gelenken und Stäben mit ihren Nummern ist optional.
- Eine Liste der Stäbe mit den in Ihnen wirkenden Kräften

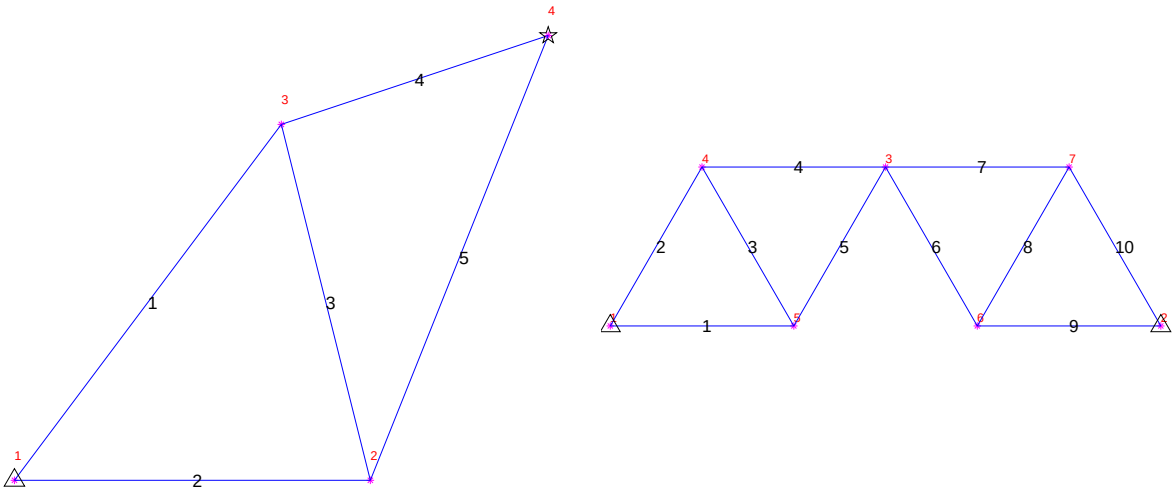

Abbildung 1: Beispiel für die Visualisierung von Fachwerken. Das linke Fachwerk entspricht dem Beispiel aus Abschnitt 3.8.2 der Vorlesung, das rechte dem Fachwerk aus Figure 11.6 aus dem Buch “Ingenieurmechanik I” von Sayir, Dual und Kaufmann (Vieweg/Teubner 2008). Gelenke sind durch $*$ in der Farbe Magenta bezeichnet, Stäbe durch blaue Linien, Auflager durch schwarze Pentagramme und feste Lager durch schwarze Dreiecke.

- Eine Liste der Lagerkräfte

Der Aufruf des Fachwerksimulators ist demonstriert in Listing 1. Die Bildschirmausgabe sehen Sie in

Listing 1: Treiberfunktion für den Fachwerksimulator

```
function trussdriver
% Driver function for truss simulator
% First example: simple chevron truss with four joints and five
    links

% define truss
jointpos = [0 1 0.75 1.5;0 0 1 1.25];
links = [1 3;1 2;2 3;3 4;2 4];
supports = [1 4];
suppplanes = [0 1;0 0];
forces = [0 0 0.5 0;0 0 0.5 0];

% call simulator
disp('Simulation: simple truss from Section 3.8.2 of course');
trusssimulator(jointpos,links,supports,suppplanes,forces);

clear all;

% static ideal truss model, see Sayir/Dual/Kaufmann
    Ingenieurmechanik I",
% Vieweg/Teubner 2008, Fig 11.6
h = 0.5*sqrt(3);
```

```
jointpos = [ 0 0;3 0; 1.5 h; 0.5 h; 1 0; 2 0; 2.5 h]';
links = [1 5; 1 4; 4 5; 3 4; 3 5; 3 6;3 7; 6 7; 2 6; 2 7];
supports = [1 2];
suppplanes = [0 0;0 0]';
forces = [0 0;0 0;0 0;0 -1;0 0; 0 0;-sqrt(3) 0]';

% call simulator
disp('Truss from Figure 11.6 of book "Ingenieurmechanik I"');
trusssimulator(jointpos,links,supports,suppplanes,forces);
```

Listing 2: Bildschirmusgabe von `trussdriver`

```
>> trussdriver
Simulation: simple truss from Section 3.8.2 of course
Force in link between joints [1,3] (direction
   (6.000000e-01,8.000000e-01)) = 0.625000
Force in link between joints [1,2] (direction (1,0)) =
   0.025000
Force in link between joints [2,3] (direction
   (-2.425356e-01,9.701425e-01)) = -0.039645
Force in link between joints [3,4] (direction
   (9.486833e-01,3.162278e-01)) = -0.121626
Force in link between joints [2,4] (direction
   (3.713907e-01,9.284767e-01)) = 0.041424
Force (Cartesian coordinates) on fixed support at (0,0) =
   (-0.400000,-0.500000)
Normal force on sliding support at
   (1.500000e+00,1.250000e+00) = -0.100000
Truss from Figure 11.6 of book "Ingenieurmechanik I"
Force in link between joints [1,5] (direction (1,0)) =
   -0.384900
Force in link between joints [1,4] (direction
   (5.000000e-01,8.660254e-01)) = -1.539601
Force in link between joints [4,5] (direction
   (5.000000e-01,-8.660254e-01)) = 0.384900
Force in link between joints [3,4] (direction (-1,0)) =
   -0.962250
Force in link between joints [3,5] (direction
   (-5.000000e-01,-8.660254e-01)) = -0.384900
Force in link between joints [3,6] (direction
   (5.000000e-01,-8.660254e-01)) = 0.384900
Force in link between joints [3,7] (direction (1,0)) =
   -1.347151
Force in link between joints [6,7] (direction
   (5.000000e-01,8.660254e-01)) = -0.384900
Force in link between joints [2,6] (direction (-1,0)) =
   0.384900
Force in link between joints [2,7] (direction
```

```
   (-5.000000e-01,8.660254e-01)) = 0.384900
Force (Cartesian coordinates) on fixed support at (0,0) =
   (1.154701,1.333333)
Force (Cartesian coordinates) on fixed support at (3,0) =
   (0.577350,-0.333333)
```

Von einem ETH-Ingenieur wird erwartet, dass er solche Implementierungsaufgaben problemlos meistert!

2 Teilprobleme

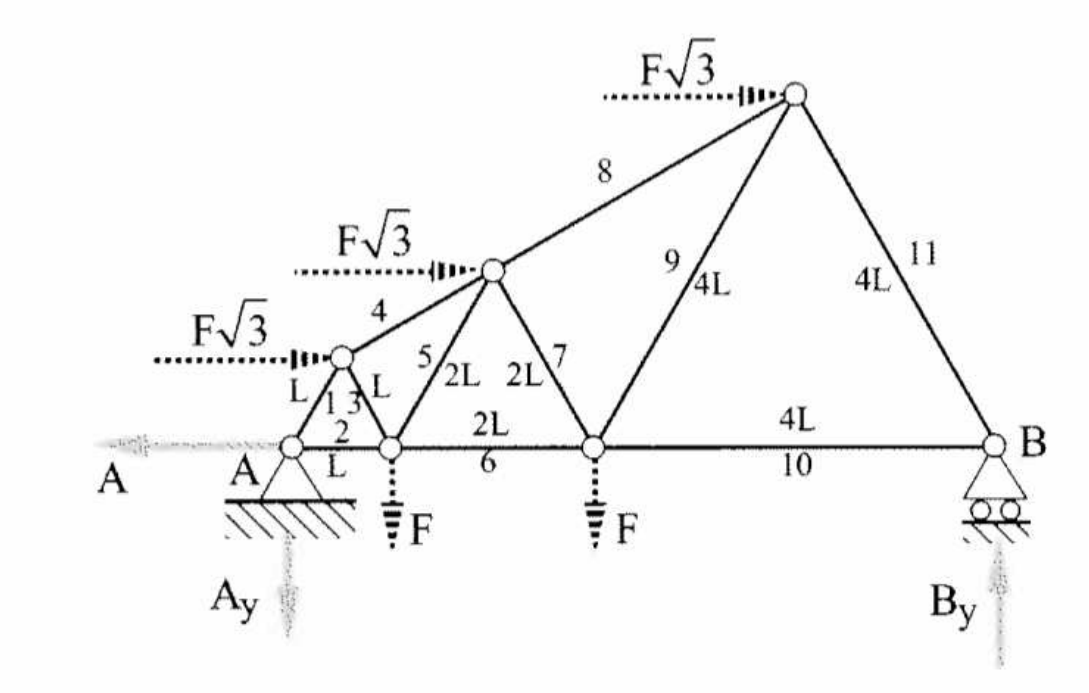

Abbildung 2: Ideales statisch bestimmtes Fachwerk aus Fig. 11.9 von [1]

1. Schreiben Sie ein MATLAB-Programm, das die Eingabematrizen und -Vektoren für `trusssimulator` für das Fachwerk aus Abbildung 2 initialisiert. Hierbei darf $F = 1\mathrm{N}$ angenommen werden, ebenso wie $L = 1\mathrm{m}$.

2. Mit welchen MATLAB-Anweisungen lässt sich aus dem Argument `jointpos` die Anzahl der Gelenke extrahieren?

3. Schreiben Sie einen kurzen MATLAB-Code, der aus den Argumenten von `trusssimulator` die Anzahl r von festen Lagern und die Anzahl q von Auflagern berechnet.

   Hinweis: In diesem Fall ist die Abfrage, ob ein Vektor gleich dem Nullvektor ist, problemlos, da von den entsprechenden Eingabedaten erwartet werden kann, dass sie keine Rundungsfehler aufweisen.

4. Welche MATLAB-Anweisungen sind nötig, um aus den Eingaben von `trusssimulator` eine $2 \times s$-Matrix `d` zu erzeugen, deren Spalten Einheitsvektoren (d.h. Vektoren der Euklidischen Länge 1) enthalten, die in Richtung der Stäbe zeigen.

5. Der `trusssimulator` soll mit einer Fehlermeldung abbrechen, wenn das Fachwerk nicht statisch bestimmt sein kann. Welche MATLAB-Anweisungen realisieren dies?

   Hinweis. Daten, die bereits in den vorherigen Teilaufgaben berechnet worden sind, können natürlich verwendet werden.

6. Das Fachwerkproblem kann zurückgeführt werden auf ein lineares Gleichungssystem $\mathbf{A}\mathbf{x} = \mathbf{b}$. Beschreiben Sie genau, welche Konventionen Sie anwenden, um Kräften die Spalten der Koeffizientenmatrix $\mathbf{A}$ zuzuordnen, und um die Kraftgleichgewichtsbedingungen mit den Zeilen der Koeffizientenmatrix $\mathbf{A}$ zu identifizieren. Was sind die Unbekannten x_i des linearen Gleichungssystems, repräsentiert durch den Spaltenvektor $\mathbf{x}$?

7. Mit welchen MATLAB-Anweisungen können Sie den Rechte-Seite-Vektor $\mathbf{b}$ aufstellen?

8. Nun schreiben Sie bitte einen MATLAB-Code, der unter Verwendung der in den vorherigen Teilaufgaben berechneten Daten, insbesondere der Matrix `d`, die Koeffizientenmatrix $\mathbf{A}$ für das Fachwerkproblem im `trusssimulator` aufstellt.

9. Implementieren Sie die Visualisierung des Fachwerks wie in Abbildung 1 (ohne Text) mit Hilfe des `plot`-Kommandos von MATLAB.

10. Berechnen Sie nun mit Ihrer Implementierung von `trusssimulator` die Kräfte im Fachwerk aus Abbildung 2. Zur Kontrolle: die Kraft auf Lager A ist $(3\sqrt{3}\mathrm{N}, \frac{1}{14}\mathrm{N})$, siehe [1, Abschnitt 11.3].

Literatur

[1] M. SAYIR, J. DUAL, AND S. KAUFMANN, *Ingenieurmathematik*, Vieweg Studium, Vieweg+Teubner, 2nd ed., 2008.

Listing 3: Fachwersimulator

```
function [forces,k,s,r,q] =
   trusssimulator(jointpos,links,supports,suppplanes,forces)
% MATLAB function for computing the forces in a truss (Fachwerk)
%
% Function arguments:
%
% 'jointpos' is a 2 x k matrix whos column contain the locations of
% the n joints
% 'links' is a s x 2 matrix whose rows specify pairs of joints
% connected by links
% 'supports' is an integer vector of length r which gives the index
% numbers of joints with external support
```

```
% 'supplanes' is an 2 x r matrix and provides the normal vectors for
% the sliding planes of supports. If the normal is (exactly!) zero we
% deal with a fixed support.
% 'forces' is a 2 x k matrix whose columns give the coordinates of the
% external forces on the joints.
%
% Return values:
%
% k = number of joints, s = number of links, r = number of fixed
% supports, q = number of sliding supports
%
% 'forces' is a vector of force components. The first s components
% contain the forces in the links, the other components give the
% forces acting on supports; for fixed supports this is a two
% component force, for sliding supports the normal force

k = size(jointpos,2); % Number of joints (= number of columns of 'jointpos')
s = size(links,1);    % Number of links ( = number of rows of 'links')
% Number of fixed supports = No. of zero normal vectors in
% 'suppplanes'
r = sum(sum(suppplanes.*suppplanes) == 0);
% Number of sliding supports
q = length(supports) - r;

% Check necessary conditions for well defined (statisch bestimmt) truss
if (2*k-s-2*r-q ~= 0), error('No well defined truss'); end

% Compute the directional vectors of the links
d = jointpos(:,links(:,2)) - jointpos(:,links(:,1)); % A 2 x s matrix
% Normalized link directions
norms = sqrt(sum(d.*d)); d = d./[norms;norms];

% Build the truss matrix from two parts; Part I stored in the variable
% A1 encodes the equilibrium conditions at the joints effected by the
% link forces. It has size (2*k) x s
A1 = zeros(2*k,s);

% The matrix can be initialized efficiently by traversing the links
% instead of looping over the joints: this trick is called assembly.
for l=1:s
  % j1 and j2 contain the indices of the joints connected by the l-th link
  j1 = links(l,1); j2 = links(l,2);
```

```
54    % the l-th column of the matrix corresponds to the force in the
        l-th
55    % link the matrix rows 2j-1 and 2j correspond to the j-th joint;
56    % rows with odd index represent the x-components of forces, those
57    % with even indices the y-components
58    A1([2*j1-1,2*j1],l) = d(:,l);
59    A1([2*j2-1,2*j2],l) = -d(:,l);
60  end
61
62  % Part II of the matrix stored in A2 includes the forces at
63  % supports. It has size (2*k) x (q+2*r)), with two columns (two unit
64  % vectors) associated with each fixed supports and a single column,
65  % whose two possibly nonzero entries are the components of the
      normal
66  % vector, belonging to each sliding support.
67  A2 = zeros(2*k,q+2*r);
68  s_idx = 1; % Column index for current support
69  % Add equations for supports; loop over all supports
70  for l=1:(r+q)
71    % Index of supported joint
72    j = supports(l);
73    % Decide whether we deal with a sliding support Note: here a test
        for
74    % equality with zero is admissible, because zero normals will not
        be
75    % the result of floating point computations and zero is contained
        in
76    % the set of machine numbers
77    if (norm(suppplanes(:,l)) ~= 0)
78      % Sliding support: two entries of a single column are
          initialized with
79      % the components of the normal vector for the l-th sliding
          support
80      A2([2*j-1,2*j],s_idx) = suppplanes(:,l);
81      s_idx = s_idx + 1; % A single column has been initialized
82    else
83      % A fixed support: The force on the l-th support (fixed) spawns
84      % columns s+2*l-1 and s+2*l of the matrix. These are unit
          vectors
85      A2(2*j-1,s_idx) = 1; A2(2*j,s_idx+1) = 1;
86      % Two columns have been initialized! Increment index.
87      s_idx = s_idx + 2;
88    end
89  end
90
91  % Assemble complete system matrix by concatenating A1 and A2
      horizontally
92  A = [A1,A2];
93  % Right hand side vector taking into account forces on the joints.
94  % The MATLAB reshape command stacks the vectors given as the columns
95  % of 'forces' on top of each other, see the MATLAB documentation
```

```
b = -reshape(forces,2*k,1);

% Compute components of all forces. The first s components of x
% contains the forces in the links, the other components give the
% forces acting on supports; for fixed supports this is a two
% component force, for sliding supports the normal force
jf = linsolve(A,b);

% Draw truss
figure('name','Truss');
% Mark joints by magenta star
plot(jointpos(1,:),jointpos(2,:),'m*'); hold on;
ax = axis; axis off; axis equal;
dx = ax(2)-ax(1); dy = ax(4) - ax(3);
% Annotate joints
for l=1:k
  text(jointpos(1,l),jointpos(2,l)+0.05*dy,num2str(l),'color','r');
end

% Visualize links
for l=1:s
  % Draw blue line for a link
  plot(jointpos(1,links(l,:)),jointpos(2,links(l,:)),'b-');
  % Annotate links with their numbers
  mp = 0.5*(jointpos(:,links(l,1))+jointpos(:,links(l,2)));
  text(mp(1),mp(2),num2str(l),'color','k','fontsize',14);
end

% Visualize supports
for l=1:(r+q)
  j = supports(l); % Index of supported joint
  if (norm(suppplanes(:,l)) ~= 0)
    % Pentagram for sliding supports
    plot(jointpos(1,j),jointpos(2,j),'pk','markersize',14);
  else
    % Black triangle for fixed support
    plot(jointpos(1,j),jointpos(2,j),'^k','markersize',14);
  end
end

% Textual output of forces
% List forces in links
for l=1:s
  fprintf('Force in link between joints [%d,%d] (direction
    (%d,%d)) = %f\n',...
        links(l,1),links(l,2),d(1,l),d(2,l),jf(l));
end
```

```
% List forces on supports
s_idx = s+1;
for l=1:(r+q)
  j = supports(l);
  if (norm(suppplanes(:,l)) ~= 0)
    fprintf('Normal force on sliding support at (%d,%d) =
       %f\n',...
            jointpos(1,j),jointpos(2,j),jf(s_idx));
    s_idx = s_idx + 1;
  else
    fprintf('Force (Cartesian coordinates) on fixed support
       at (%d,%d) = (%f,%f)\n',...
         jointpos(1,j),jointpos(2,j),jf(s_idx),jf(s_idx+1));
    s_idx = s_idx + 2;
  end
end
```