Ostfalia Hochschule für angewandte Wissenschaften

Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476

1 FEM-Programm

1.1 Main()

```
%INIT
clear();                                                                    %Variablen löschen
clc();                                                                      %Konsole löschen
close all;                                                                  %Plots schließen

%MAIN()----------------------------------------------------------------------------------------------
    [iANZAHL_KNOTEN,aKNOTENKOORDINATEN,aSTABELEMENTE,aKNOTENLASTEN,aLAGERBEDINGUNGEN,sDATEINAME]=Datensatz_Einlesen();
    [aALPHA,aL,aEA,aSTABVERBINDUNGEN]=Stab_Eigenschaften(aSTABELEMENTE,aKNOTENKOORDINATEN);
    clear aSTABELEMENTE;
    aK_ges=Gesamtsteifigkeitsmatrix_Bilden(iANZAHL_KNOTEN,aSTABVERBINDUNGEN,aALPHA,aL,aEA,aLAGERBEDINGUNGEN);
    aF_ges=Elementlastvektor_Bilden(iANZAHL_KNOTEN,aKNOTENLASTEN);
    aU_ges=Cholesky_Verfahren(aK_ges,aF_ges);
    aN_ges=Kraftvektor_Ermitteln(aSTABVERBINDUNGEN,aU_ges,aALPHA,aL,aEA);
    Ergebnisse_Ausgeben(sDATEINAME,aU_ges,aN_ges);
    clear sDATEINAME;
    Verformung_Plotten(aSTABVERBINDUNGEN,aKNOTENKOORDINATEN,aU_ges,aN_ges,aL,aEA);
    clear aKNOTENKOORDINATEN;
%----------------------------------------------------------------------------------------------------
```

Datensatz_Einlesen()

Liest die Textdatei mit den Daten ein	Speichert die Daten in Arrays zur Weiterverwendung ab und prüft Vollständigkeit

Stab_Eigenschaften()

Benötigt alle Parameter-Arrays der Stäbe und Knoten	errechnet dLagewinkel; Länge und Steifigkeit der Stäbe

Gesamtsteifigkeitsmatrix_Bilden()

Benötigt Lagewinkel; Länge; Steifigkeit und zugehörige Knoten der Stäbe + Lagerung	Bildet die Matrix K des Gleichungssystems inklusive Randbedingungen $K \cdot u = f$

Elementlastvektor_Bilden()

Benötigt die Punkte und Richtungen der Lagerfixierungen	Erstellt den Ergebnisvektor f des Gleichungssystems $K \cdot u = f$

Cholesky_Verfahren()

Benötigt die symmetrische Matrix K und den Ergebnisvektor f	Errechnet den Verschiebungsvektor u des Gleichungssystems $K \cdot u = f$

Kraftvektor_Ermitteln()

Benötigt Lagewinkel; Länge; Steifigkeit und zugehörige Knoten der Stäbe + Verschiebungsvektor u	Errechnet aus den Spannungsmatrizen der Stabelemente den Normalkraftvektor N

Ergebnisse_Ausgeben()

Benötigt den Verschiebungsvektor u und den Normalkraftvektor N	Schreibt die eingelesene Textdatei in eine Ausgabetextdatei und hängt die Ergebnisse der Verschiebungen un Normalkräfte am Ende an

Verformung_Plotten()

Benötigt den Verschiebungsvektor u; den Normalkraftvektor N; die Längen und Knotenverbindungen der Stäbe und die Knotenkoordinaten	Zeichnet das unverformte Fachwerk und unterlagert ein übertrieben verformtes System mit eingefärbten Stäben nach N (0-100% = grün-rot)

1.2 Datensatz_Einlesen()

```
function [iANZAHL_KNOTEN,aKNOTEN,aSTABELEMENTE,aKNOTENLASTEN,aLAGERBEDINGUNGEN,sDATEINAME] = Datensatz_Einlesen()
    %KONFIGURATION------------------------------------------------------------------------------------------
    sSOLL(1)="Steuerdaten:";
    sSOLL(2)="Knoten:";
    sSOLL(3)="Stabelemente:";
    sSOLL(4)="Knotenlasten:";
    sSOLL(5)="Lagerbedingungen:";

    %ÖFFNEN_DER_TEXTDATEI-----------------------------------------------------------------------------------
    sDATEINAME = input('Geben Sie den Dateinamen mit den Steuerdaten ein (Dateityp: ".dat")\n','s');
    sDATEINAME = strcat(sDATEINAME,".dat");
    fhTEXTDATEI = fopen(sDATEINAME,'r');
    if (fhTEXTDATEI < 1)
        fprintf('%s konnte im Arbeitsverzeichnis nicht gefunden werden\nProgramm wird beendet\n',sDATEINAME);
        return;
    else
        fprintf('%s wurde geöffnet\n',sDATEINAME);
        fprintf('//---------------------------------\n');
    end


    %EINLESEN_DER_STEUERDATEN-------------------------------------------------------------------------------
    %Steuerdaten
    frewind(fhTEXTDATEI);%Pointer zurücksetzen (beim ersten Mal unnötig -> egal)
    sZEILE="";
    while(sZEILE~=sSOLL(1))
        sZEILE=fgetl(fhTEXTDATEI);
        if(sZEILE==-1)
            fprintf('Formatierung in %s fehlerhaft!\n%s erwartet!\n',sDATEINAME,sSOLL(1));
            fprintf('//---------------------------------\n');
            return;
        end
    end
    for(i=1:1:3)%Weglesen der ***; Tabellenüberschriften und ---,-----
        sZEILE=fgetl(fhTEXTDATEI);
    end
    aSPEICHER=fscanf(fhTEXTDATEI,'%u',4);
    aSTEUERDATEN(1,1:4)=aSPEICHER.';

    %Knoten
    frewind(fhTEXTDATEI);
    sZEILE="";
    while(sZEILE~=sSOLL(2))
        sZEILE=fgetl(fhTEXTDATEI);
        if(sZEILE==-1)
            fprintf('Formatierung in %s fehlerhaft!\n%s erwartet!\n',sDATEINAME,sSOLL(2));
            fprintf('//---------------------------------\n');
            return;
        end
    end
    for(i=1:1:3)
        sZEILE=fgetl(fhTEXTDATEI);
    end
    i=1;
    while(1)
        iPOINTER=ftell(fhTEXTDATEI);
        sZEILE=fgetl(fhTEXTDATEI);
        iMERKER=0;
        for(j=1:1:size(sSOLL,2))
            if(sZEILE==sSOLL(j))
                iMERKER=1;
            end
        end
        if(iMERKER==1||strrep(strrep(sZEILE," ",""),"\t","")==""||sZEILE=="EOD")
            break;
        else
            fseek(fhTEXTDATEI,iPOINTER,'bof'); %Pointer zurücksetzen, weil ja Zeile zum Test schon eingelesen wurde
            aSPEICHER=fscanf(fhTEXTDATEI,'%f',3);
            aKNOTEN(i,1:3)=aSPEICHER.';
            fseek(fhTEXTDATEI,ftell(fhTEXTDATEI)+2,'bof'); %Überlesen des Zeilenumbruchs
        end
        i=i+1;
    end

    %Stabelemente
    frewind(fhTEXTDATEI);
    sZEILE="";
    while(sZEILE~=sSOLL(3))
        sZEILE=fgetl(fhTEXTDATEI);
        if(sZEILE==-1)
            fprintf('Formatierung in %s fehlerhaft!\n%s erwartet!\n',sDATEINAME,sSOLL(3));
            fprintf('//---------------------------------\n');
            return;
        end
    end
    for(i=1:1:3)
        sZEILE=fgetl(fhTEXTDATEI);
    end
    i=1;
    while(1)
        iPOINTER=ftell(fhTEXTDATEI);
        sZEILE=fgetl(fhTEXTDATEI);
        iMERKER=0;
        for(j=1:1:size(sSOLL,2))
            if(sZEILE==sSOLL(j))
                iMERKER=1;
            end
        end
```

```
    if(iMERKER==1||strrep(strrep(sZEILE," ",""),"\t","")==""||sZEILE=="EOD")
        break;
    else
        fseek(fhTEXTDATEI,iPOINTER,'bof');
        aSPEICHER=fscanf(fhTEXTDATEI,'%f',5);
        aSTABELEMENTE(i,1:5)=aSPEICHER.';
        fseek(fhTEXTDATEI,ftell(fhTEXTDATEI)+2,'bof');
    end
    i=i+1;
end

%Knotenlasten
frewind(fhTEXTDATEI);
sZEILE="";
while(sZEILE~=sSOLL(4))
    sZEILE=fgetl(fhTEXTDATEI);
    if(sZEILE==-1)
        fprintf('Formatierung in %s fehlerhaft!\n%s erwartet!\n',sDATEINAME,sSOLL(4));
        fprintf('//--------------------------------\n');
        return;
    end
end
for(i=1:1:3)
    sZEILE=fgetl(fhTEXTDATEI);
end
i=1;
while(1)
    iPOINTER=ftell(fhTEXTDATEI);
    sZEILE=fgetl(fhTEXTDATEI);
    iMERKER=0;
    for(j=1:1:size(sSOLL,2))
        if(sZEILE==sSOLL(j))
            iMERKER=1;
        end
    end
    if(iMERKER==1||strrep(strrep(sZEILE," ",""),"\t","")==""||sZEILE=="EOD")
        break;
    else
        fseek(fhTEXTDATEI,iPOINTER,'bof');
        aSPEICHER=fscanf(fhTEXTDATEI,'%f',3);
        aKNOTENLASTEN(i,1:3)=aSPEICHER.';
        fseek(fhTEXTDATEI,ftell(fhTEXTDATEI)+2,'bof');
    end
    i=i+1;
end

%Lagerbedingungen
frewind(fhTEXTDATEI);
sZEILE="";
while(sZEILE~=sSOLL(5))
    sZEILE=fgetl(fhTEXTDATEI);
    if(sZEILE==-1)
        fprintf('Formatierung in %s fehlerhaft!\n%s erwartet!\n',sDATEINAME,sSOLL(5));
        fprintf('//--------------------------------\n');
        return;
    end
end
for(i=1:1:3)
    sZEILE=fgetl(fhTEXTDATEI);
end
    i=1;
while(1)
    iPOINTER=ftell(fhTEXTDATEI);
    sZEILE=fgetl(fhTEXTDATEI);
    iMERKER=0;
    for(j=1:1:size(sSOLL,2))
        if(sZEILE==sSOLL(j))
            iMERKER=1;
        end
    end
    if(iMERKER==1||strrep(strrep(sZEILE," ",""),"\t","")==""||sZEILE=="EOD")
        break;
    else
        fseek(fhTEXTDATEI,iPOINTER,'bof');
        aSPEICHER=fscanf(fhTEXTDATEI,'%f',3);
        aLAGERBEDINGUNGEN(i,1:3)=aSPEICHER.';
        fseek(fhTEXTDATEI,ftell(fhTEXTDATEI)+2,'bof');
    end
    i=i+1;
end

%SCHLIEßEN_DER_TEXTDATEI--------------------------------------------------------------------------------
fclose(fhTEXTDATEI);

%AUFSTEIGENDE_SORTIERUNG_DER_TABELLEN_NACH_1.SPALTE----------------------------------------------------
aKNOTEN=sortrows(aKNOTEN);
aSTABELEMENTE=sortrows(aSTABELEMENTE);
aKNOTENLASTEN=sortrows(aKNOTENLASTEN);
aLAGERBEDINGUNGEN=sortrows(aLAGERBEDINGUNGEN);

%PRÜFUNG_DER_EINGELESENEN_DATEN------------------------------------------------------------------------
%Initialisierung
iANZAHL_KNOTEN=size(aKNOTEN,1);
iANZAHL_STAEBE=size(aSTABELEMENTE,1);
iANZAHL_LASTEN=size(aKNOTENLASTEN,1);
iANZAHL_RANDBEDINGUNGEN=size(aLAGERBEDINGUNGEN,1);
aSTEUERDATEN_IST(:)=[iANZAHL_KNOTEN,iANZAHL_STAEBE,iANZAHL_LASTEN,iANZAHL_RANDBEDINGUNGEN];
sWARNUNG="Scotty, wir haben ein Problem mit den ";
%Prüfung
for(i=1:1:size(aSTEUERDATEN_IST,1))
```

```
        if(aSTEUERDATEN_IST(i)<aSTEUERDATEN(i))
            fprintf('%s%s!\n',sWARNUNG,sSOLL(i+1));
            fprintf('Es gibt zu wenig Zeileneinträge (%u) unter der Überschrift "%s"\n',aSTEUERDATEN_IST(i),sSOLL(1+1));
            fprintf('Laut "%s" (%u) erwartet!',sSOLL(1),aSTEUERDATEN(i));
            fprintf('Korrigieren Sie die Datei "%s" und starten Sie das Programm erneut!\n',sDATEINAME);
            return;
        elseif(aSTEUERDATEN_IST(1)>aSTEUERDATEN(1))
            fprintf('%s%s!\n',sWARNUNG,sSOLL(i+1));
            fprintf('Es gibt zu viele Zeileneinträge (%u) unter der Überschrift "%s"\n',aSTEUERDATEN_IST(i),sSOLL(1+1));
            fprintf('Laut "%s" (%u) erwartet!\n',sSOLL(1),aSTEUERDATEN(i));
            fprintf('Korrigieren Sie die Datei "%s" und starten Sie das Programm erneut!\n',sDATEINAME);
            return;
        end
    end
end
```

Die Funktion fragt den Namen der Textdatei ab, welche die Parametrierung der Berechnung enthält und bersucht Sie zu öffnen. Sollte die Datei nicht im Arbeitsverzeichnis zu finden sind, so wird das Programm beendet.

Nach erfolgreicher Öffnung liest das Programm solange zeilenweise die Datei ein bis einer der **Stichwortzeile**n:

- „Steuerdaten:“
- „Knoten:“
- „Stabelemente:“
- „Knotenlasten:“
- „Lagerbedingungen:“

auftaucht. Aufgrund der Struktur der vorgegebenen Einlesedatei, werden stumpf zunächst 3 Zeilen überlesen. Ab der Stelle wird immer eine Zeile eingelesen. Sollte das Dateiende, statt Werten erreicht werden, so wird das Programm beendet. Sollte die Zeile (die ohne Zeilenumbruch übergeben wird) nur aus Leerzeichen und Tabulatoren bestehen, so wird das Einlesen für die **Stichwortzeile** abgebrochen. Andernfalls wird davon ausgegangen, dass es sich um Zahlenwerte handeln muss. Hierzu wird der Pointer auf die Zeile zuvor zurückgesetzt (inklusive Offset 2 für den Zeilenumbruch) und die Werte sukzessive in ein mit der **Stichwortzeile** gleichlautendes **Array** geschrieben.

Sind alle **Stichwortzeile**n abgearbeitet, wird die Textdatei geschlossen. Im nächsten Schritt erfolgt die Überprüfung der angegebenen Steuerdaten mit der Zeilenanzahl der **Stichwortzeile**n gleichlautenden Arrays. Weicht die Anzahl ab, werden entsprechende Anweisungen gegeben und das Programm beendet.

Der eingegebene Name der Textdatei wird an die main() übergeben, um für die spätere Textausgabe nocheinmal genutzt werden zu können.

1.3 Stab_Eigenschaften()

```
function [aALPHA,aL,aEA,aSTABVERBINDUNGEN] = Stab_Eigenschaften(aSTABELEMENTE,aKNOTENKOORDINATEN)
    %Koordinaten müssen in der Ausführung ausschließlich positiv sein
    %Probleme zu erwarten, wenn Knotennummerierung willkürlich geschieht
    for(i=1:1:size(aSTABELEMENTE,1))
        aALPHA(i,1)=aSTABELEMENTE(i,1);
        aL(i,1)=aSTABELEMENTE(i,1);
        aEA(i,1)=aSTABELEMENTE(i,1);
        aSTABVERBINDUNGEN(i,1)=aSTABELEMENTE(i,1);
        iANFANGSKNOTEN=min(aSTABELEMENTE(i,2),aSTABELEMENTE(i,3));
        iENDKNOTEN=max(aSTABELEMENTE(i,2),aSTABELEMENTE(i,3));
        dDELTA_X=aKNOTENKOORDINATEN(iENDKNOTEN,2)-aKNOTENKOORDINATEN(iANFANGSKNOTEN,2);
        dDELTA_Y=aKNOTENKOORDINATEN(iENDKNOTEN,3)-aKNOTENKOORDINATEN(iANFANGSKNOTEN,3);
        if(dDELTA_X==0)
            aALPHA(i,2)=atan(inf);
        elseif(dDELTA_Y==0)
            aALPHA(i,2)=0;
        else
            aALPHA(i,2)=atan(dDELTA_Y/dDELTA_X);
        end
        aL(i,2)=sqrt(dDELTA_X^2+dDELTA_Y^2);
        aEA(i,2)=aSTABELEMENTE(i,4)*aSTABELEMENTE(i,5);
        aSTABVERBINDUNGEN(i,2)=iANFANGSKNOTEN;
        aSTABVERBINDUNGEN(i,3)=iENDKNOTEN;
    end
end
```

Die Funktion führt eine Schleife für jeden Stab des Fachwerks aus. Dabei wird zunächst der Anfangknoten $k_{i,Anfang}$ und der Endknoten $k_{i,Ende}$ eines jeden Stabes ermittelt.

$$k_{i,Anfang} = \min(k_{i,1}; k_{i,2})$$

$$k_{i,Ende} = \max(k_{i,1}; k_{i,2})$$

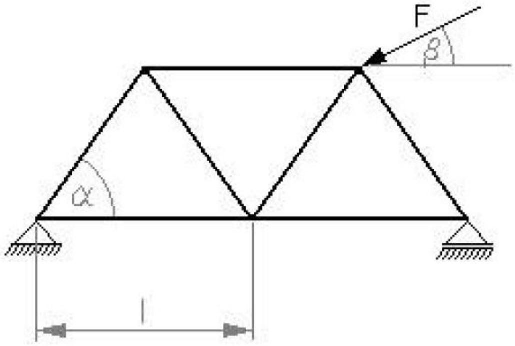
Dann wird die Länge der Stäbe und der Winkel über die Knotenkoordinaten berechnet. Sie werden in jeweils eigene Arrays abgespeichert.

$$\Delta x_i = x(k_{i,Ende}) - x(k_{i,Anfang})$$

$$\Delta y_i = y(k_{i,Ende}) - y(k_{i,Anfang})$$

$$l_i = \sqrt[2]{{\Delta x_i}^2 + {\Delta y_i}^2}$$

$$\alpha_i = \tan^{-1}\left(\frac{\Delta y}{\Delta x}\right)$$



Bildquelle: Fachwerksberechnung mit FEM I (Theorie) (Florian Grabner, HTBL-Kapfenberg) http://www.math-tech.at/Beispiele/upload/gra_FachwerkmitFEM(I).PDF

Zur weiteren Vereinfachung wird das Produkt aus E-Modul und Querschnittsfläche in einem Array abgespeichert.

$$EA_i = E_i \cdot A_i$$

Abschließend wird ein Array erzeugt in dem die Anfangsknoten $k_{i,Anfang}$ und Endknoten $k_{i,Ende}$ den Stäben zugeordnet werden. Dies führt dazu, dass das in der Funktion $Datensatz\_Einlesen()$ eingelesene Array für die Stabelemente gelöscht werden kann. Dies geschieht nachgelagert in in der $main()$.

1.4 Gesamtsteifigkeitsmatrix_Bilden()

```
function [Return] = Gesamtsteifigkeitsmatrix_Bilden(iANZAHL_KNOTEN,aSTABVERBINDUNGEN,aALPHA,aL,aEA,aLAGERBEDINGUNGEN)
    %Initialisierung des Containers (Cell-Arrays)----------------
    aVERBINDUNGSCONTAINER=cell(iANZAHL_KNOTEN);
    for(i=1:1:iANZAHL_KNOTEN)
        for(j=1:1:iANZAHL_KNOTEN)
            aVERBINDUNGSCONTAINER(i,j)={[0,0;0,0]};
        end
    end

    %Verbindungen der Knoten herstellen
    for(i=1:1:size(aSTABVERBINDUNGEN,1))
        j=aSTABVERBINDUNGEN(i,2);%Anfangsknoten
        k=aSTABVERBINDUNGEN(i,3);%Endknoten
        Speicher=Elementsteifigkeitsmatrix_Bilden(aEA,aL,aALPHA,i);
        aVERBINDUNGSCONTAINER(j,j)= {[ ...
                                        aVERBINDUNGSCONTAINER{j,j}(1,1)+Speicher(1,1), ...
                                        aVERBINDUNGSCONTAINER{j,j}(1,2)+Speicher(1,2); ...
                                        aVERBINDUNGSCONTAINER{j,j}(2,1)+Speicher(2,1), ...
                                        aVERBINDUNGSCONTAINER{j,j}(2,2)+Speicher(2,2) ...
                                    ]};
        aVERBINDUNGSCONTAINER(j,k)= {[ ...
                                        aVERBINDUNGSCONTAINER{j,k}(1,1)-Speicher(1,1), ...
                                        aVERBINDUNGSCONTAINER{j,k}(1,2)-Speicher(1,2); ...
                                        aVERBINDUNGSCONTAINER{j,k}(2,1)-Speicher(2,1), ...
                                        aVERBINDUNGSCONTAINER{j,k}(2,2)-Speicher(2,2) ...
                                    ]};
        aVERBINDUNGSCONTAINER(k,j)= {[ ...
                                        aVERBINDUNGSCONTAINER{k,j}(1,1)-Speicher(1,1), ...
                                        aVERBINDUNGSCONTAINER{k,j}(1,2)-Speicher(1,2); ...
                                        aVERBINDUNGSCONTAINER{k,j}(2,1)-Speicher(2,1), ...
                                        aVERBINDUNGSCONTAINER{k,j}(2,2)-Speicher(2,2) ...
                                    ]};
        aVERBINDUNGSCONTAINER(k,k)= {[ ...
                                        aVERBINDUNGSCONTAINER{k,k}(1,1)+Speicher(1,1), ...
```

Ostfalia Hochschule für angewandte Wissenschaften

Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476

```
                                aVERBINDUNGSCONTAINER{k,k}(1,2)+Speicher(1,2); ...
                                aVERBINDUNGSCONTAINER{k,k}(2,1)+Speicher(2,1), ...
                                aVERBINDUNGSCONTAINER{k,k}(2,2)+Speicher(2,2) ...
                            ]};
    end

    %Einbauen der Randbedingungen
    for(i=1:1:size(aLAGERBEDINGUNGEN,1))
        iKNOTEN=aLAGERBEDINGUNGEN(i,1);
        iTRANSLATIONSSPERRE=aLAGERBEDINGUNGEN(i,2);% 1 ist x, 2 ist y, gleiches gilt für die Cell-Array-Logik
        for(j=1:1:iANZAHL_KNOTEN)%Streichen in den Spalten des Containers
            aVERBINDUNGSCONTAINER{j,iKNOTEN}(1,iTRANSLATIONSSPERRE)=0;
            aVERBINDUNGSCONTAINER{j,iKNOTEN}(2,iTRANSLATIONSSPERRE)=0;
        end
        for(k=1:1:iANZAHL_KNOTEN)%Streichen in den Zeilen des Containers
            aVERBINDUNGSCONTAINER{iKNOTEN,k}(iTRANSLATIONSSPERRE,1)=0;
            aVERBINDUNGSCONTAINER{iKNOTEN,k}(iTRANSLATIONSSPERRE,2)=0;
        end
        aVERBINDUNGSCONTAINER{iKNOTEN,iKNOTEN}(iTRANSLATIONSSPERRE,iTRANSLATIONSSPERRE)=1;
    end

    %Explosion des Cell-Arrays in ein Array
    for(i=2:2:2*iANZAHL_KNOTEN)
        for(j=2:2:2*iANZAHL_KNOTEN)
            for(k=1:1:2)
                for(l=1:1:2)
                    Return(i+k-2,j+l-2)=aVERBINDUNGSCONTAINER{i/2,j/2}(k,l);
                end
            end
        end
    end
end
```

Im ersten Schritt wird ein Cell-Array erstellt, dass auf der obersten Ebene so hoch und breit wie die Anzahl der Knoten im System sind. Jede Zelle des Cell-Arrays wird mit einer null-gefüllten 2x2-Matrix initialisiert.

Im nächsten Schritt wird die Gesamtsteifigkeitsmatrix KK_g ohne die Lagerbedingungen aufgebaut. Die Elementsteifigkeitsmatrizen werden wie folgt in das Cell-Array verteilt. Entscheidend für die Aufteilung jeder Elementsteifigkeitsmatrix eines Stabes ist der Anfangsknoten und der Endknoten. So wird die grüne Elementsteifigkeitsmatrix für einen Stab, der mit den Knoten 1 und 3 verbunden ist, wie in der Abbildung in das mit 0 initialisierte Cell-Array aufsummiert.

$$
\begin{array}{c}
\begin{array}{ccccc c} \quad 1 \quad & \quad 2 \quad & \quad 3 \quad & \quad 4 \quad & \quad 5 \quad & \text{Knoten} \end{array} \\
KK_g = \begin{bmatrix}
\boxed{KE_1} + \boxed{KE_2} & \boxed{-KE_1} & \boxed{-KE_2} & 0 & 0 \\
\boxed{-KE_1} & \boxed{KE_1} + KE_3 + KE_4 & -KE_3 & -KE_4 & 0 \\
\boxed{-KE_2} & -KE_3 & \boxed{KE_2} + KE_3 + KE_5 + KE_6 & -KE_5 & -KE_6 \\
0 & -KE_4 & -KE_5 & KE_4 + KE_5 + KE_7 & -KE_7 \\
0 & 0 & -KE_6 & -KE_7 & KE_6 + KE_7
\end{bmatrix}
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\end{array}
$$

$$KE_i = \frac{E_i \cdot A_i}{l_i} \cdot \begin{pmatrix} \cos^2(\alpha_i) & \sin(\alpha_i) \cdot \cos(\alpha_i) \\ \sin(\alpha_i) \cdot \cos(\alpha_i) & \sin^2(\alpha_i) \end{pmatrix}$$

Bildquelle: Fachwerksberechnung mit FEM I (Theorie) (Florian Grabner, HTBL-Kapfenberg) http://www.math-tech.at/Beispiele/upload/gra_FachwerkmitFEM(I).PDF

Nachfolgend werden die Randbedingungen der Lagerung in die Gesamtsteifigkeitsmatrix eingebaut. Gesperrte Translationen eines Knotens führen dazu, dass die zur Translation dazugehörigen Spalten und Zeilen zu 0 gesetzt werden. Am Schnittpunkt der Spalten und Zeilen (dem Diagonalenelement) wird die Zelle zu 1 gesetzt.

Schlussendlich wird das Cellarray zu einer Matrix, der Gesamtsteifigkeitsmatrix, explodiert.

1.4.1 Elementsteifigkeitsmatrix_Bilden()

```
function [Return] = Elementsteifigkeitsmatrix_Bilden(aEA,aL,aALPHA,stabindex)
    Return=zeros(2,2);
    %Funktion nur verlässlich wenn Arrayindex mit Stabindex übereinstimmt
    Return(1,1)=((cos(aALPHA(stabindex,2))^2)                            *aEA(stabindex,2))/aL(stabindex,2);
    Return(1,2)=( sin(aALPHA(stabindex,2))*cos(aALPHA(stabindex,2))  *aEA(stabindex,2))/aL(stabindex,2);
    Return(2,1)=( sin(aALPHA(stabindex,2))*cos(aALPHA(stabindex,2))  *aEA(stabindex,2))/aL(stabindex,2);
    Return(2,2)=((sin(aALPHA(stabindex,2))^2)                            *aEA(stabindex,2))/aL(stabindex,2);
    dSCHRANKE=0.00000001;
    if(abs(Return(1,1))<dSCHRANKE)
        Return(1,1)=0;
    end
    if(abs(Return(1,2))<dSCHRANKE)
        Return(1,2)=0;
    end
    if(abs(Return(2,1))<dSCHRANKE)
        Return(2,1)=0;
    end
    if(abs(Return(2,2))<dSCHRANKE)
        Return(2,2)=0;
    end
end
```

Die Elementsteifigkeitsmatrix bereitet folgende Rechnung für die Implementierung in die Gesamtsteifigkeitsmatrix vor:

$$KE_i = \frac{E_i \cdot A_i}{l_i} \cdot \begin{pmatrix} \cos^2(\alpha_i) & \sin(\alpha_i) \cdot \cos(\alpha_i) \\ \sin(\alpha_i) \cdot \cos(\alpha_i) & \sin^2(\alpha_i) \end{pmatrix}$$

Abschließend werden sehr kleine Werte um 0 herum zu 0 gesetzt. Leider führt die Matlab-interne Rechnung des Sinus aus $\frac{\pi}{2}$ zu Werten knapp größer als 0. Diese unsaubere Lösung wurde eingeführt, um die Gesamtsteifigkeitsmatrix besser debuggen zu können. Sie wird durch die falschen Ergebnisse sonst verfälscht und schlecht lesbar.

1.5 Elementlastvektor_Bilden()

```
function [Return] = Elementlastvektor_Bilden(iANZAHL_KNOTEN,aKNOTENLASTEN)
    Return=zeros(2*iANZAHL_KNOTEN,1);
    for(i=1:1:size(aKNOTENLASTEN,1))
        iKNOTEN=aKNOTENLASTEN(i,1);
        iRICHTUNG=aKNOTENLASTEN(i,2);%1 = x-Richtung, 2 = y-Richtung
        Return(2*iKNOTEN+iRICHTUNG-2)=aKNOTENLASTEN(i,3);
    end
end
```

In dieser Funktion werden ie angegebenen Lasten in folgender Vektorform f aufgereiht:

$$f = \begin{pmatrix} F_{1,x} \\ F_{1,y} \\ \vdots \\ F_{i,x} \\ F_{i,y} \end{pmatrix}$$

1.6 Cholesky_Verfahren()

```
function [Return] = Cholesky_Verfahren(aA,aB)
    %PRUEFUNG-------------------------------------------------------------------------
    iANZAHL_GLEICHUNGEN = size(aA,1);               %Gibt die Anzahl an Zeilen zurück (Parameter 1)
    iANZAHL_WERTE = size(aA,2);                     %Gibt die Anzahl der Spalten zurück
    if(iANZAHL_GLEICHUNGEN<iANZAHL_WERTE)
       fprintf("Scotty, wir haben ein Problem!\n");
       fprintf("Es gibt weniger Gleichungen als Unbekannte!\n");
       return;                                      %Hoffe das beendet das Skript vorzeitig
    end
    if(issymmetric(aA))                             %ist das gleiche wie if(aA==aA.')
       fprintf("Matrix ist symmetrisch\n");
    else
       fprintf("Scotty, wir haben ein Problem!\n");
       fprintf("Matrix ist nicht symmetrisch\n");
```

```
        Return=Gauss_Verfahren(aA,aB);
        return;
    end

    %R_ZERLEGUNG-----------------------------------------------------------------------
    aR=zeros(iANZAHL_GLEICHUNGEN,iANZAHL_WERTE);            %Initialisierung einer 0-Matrix
    for(i=1:1:iANZAHL_GLEICHUNGEN)
        for(j=i:1:iANZAHL_WERTE                             %die Spalte j startet mit jedem Durchlauf bei dem Wert der Zeile i
            %Bearbeitung der Diagonal-Lümmel
            Zwischenspeicher=0;
            for(k=1:1:i-1)
                Zwischenspeicher=Zwischenspeicher+(aR(k,i)^2);
            end
            aR(i,i)=(aA(i,i)-Zwischenspeicher)^0.5;
            if((aA(i,i)-Zwischenspeicher) <= 0)
                fprintf("Matrix ist nicht positiv definit!\n");
                fprintf("Geprüft an: R(%u,%u) = %f\n",i,i,(aA(i,i)-Zwischenspeicher));
                Return=Gauss_Verfahren(aA,aB);
                return;                                     %Ich hoffe das Programm bricht dadurch ab
            else
                fprintf("Postive Definitheit an Stelle: R(%u,%u) = %f erfolgreich geprüft\n",i,i,(aA(i,i)-Zwischenspeicher));
            end
            %Bearbeitung der Nicht-Diagonal-Lümmel
            Zwischenspeicher = 0;
            for(k=1:1:i-1)
                Zwischenspeicher=Zwischenspeicher+(aR(k,i)*aR(k,j));
            end
            aR(i,j)=(aA(i,j)-Zwischenspeicher)/aR(i,i);
        end
    end
    aRT=aR';                                                %Erzeugung der transponierten Matrix -> alternativ auch aRT=aR.';

    %VORWÄRTSAUFLÖSUNG-----------------------------------------------------------------
    for(i=1:1:iANZAHL_GLEICHUNGEN)
        Zwischenspeicher=0;
        for(j=1:1:i-1)
            Zwischenspeicher=Zwischenspeicher+(aRT(i,j)*aY(j));
        end
        aY(i)=(aB(i)-Zwischenspeicher)/aRT(i,i);
    end

    %RÜCKWÄRTSAUFLÖSUNG----------------------------------------------------------------
    for(i=iANZAHL_GLEICHUNGEN:-1:1)
        Zwischenspeicher=0;
        for(j=i+1:1:iANZAHL_WERTE)
            Zwischenspeicher=Zwischenspeicher+(aR(i,j)*aX(j));
        end
        aX(i)=(aY(i)-Zwischenspeicher)/aR(i,i);
    end
    aX=aX.';
    Return=aX(1:end);
end
```

Das Cholesky-Verfahren wurde gewählt, weil die Gesamtsteifigkeitsmatrizen symmetrisch sind. Zusammen mit dem Elementlastvektor (Ergebnisvektor), kann die Verschiebung u errechnet werden.

$$K \cdot u = f$$

Sollte die Matrix nicht symmetrisch sein oder an einer Stelle nicht positiv definit sein, wird versucht das Gleichungssystem per Gauß-Verfahren zu lösen.

1.7 Kraftvektor_Ermitteln()

```
function [Return] = Kraftvektor_Ermitteln(aSTABVERBINDUNGEN,aU_ges,aALPHA,aL,aEA);
    for(i=1:1:size(aSTABVERBINDUNGEN,1))
        Anfangsknoten=aSTABVERBINDUNGEN(i,2);
        Endknoten=aSTABVERBINDUNGEN(i,3);
        Zwischenspeicher(1:2,1:2)=[1,-1;-1,1];
        aK=(aEA(i,2)/aL(i,2))*Zwischenspeicher;
        aT(1:2,1:4)=[cos(aALPHA(i,2)),sin(aALPHA(i,2)),0,0;0,0,cos(aALPHA(i,2)),sin(aALPHA(i,2))];
        dSCHRANKE=0.00000001;
        if(abs(aT(1,1))<dSCHRANKE)
            aT(1,1)=0;
        end
        if(abs(aT(1,2))<dSCHRANKE)
            aT(1,2)=0;
        end
        if(abs(aT(2,3))<dSCHRANKE)
            aT(2,3)=0;
        end
        if(abs(aT(2,4))<dSCHRANKE)
            aT(2,4)=0;
        end
        aU(1:2,1)=aU_ges((2*Anfangsknoten)-1:(2*Anfangsknoten),1);
        aU(3:4,1)=aU_ges((2*Endknoten)-1:(2*Endknoten),1);
        Zwischenspeicher=aK*aT*aU;
        Zwischenspeicher=Zwischenspeicher';
        dSCHRANKE=0.00000001;
        if(abs(Zwischenspeicher(1,1))<dSCHRANKE)
            Zwischenspeicher(1,1)=0;
        end
        if(abs(Zwischenspeicher(1,2))<dSCHRANKE)
```

```
            Zwischenspeicher(1,2)=0;
        end
        Return(i,1)=i;
        Return(i,2:3)=Zwischenspeicher;
    end
end
```

Zur Ermittlung der Normalkräfte wird zunächst die Elementsteifigkeitsmatrix eines Stabes im lokalen Koordinatensystem gebildet:

$$K_i = \frac{E_i \cdot A_i}{l_i} \cdot \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Außerdem wird eine Transformationsmatrix für das globale Koordinatensystem benötigt:

$$T_i = \begin{pmatrix} \cos(\propto_i) & \sin(\propto_i) & 0 & 0 \\ 0 & 0 & \cos(\propto_i) & \sin(\propto_i) \end{pmatrix}$$

Da in Matlab-internen Berechnungen der Sinus von 90° zu einem Ergebnis knapp über 0 führt, wird das Ergebnis um eine enge Schranke auf 0 gesetzt. Dies ist unsauber, aber die Werte verfälschen systematisch die Lösung. Hier sollte nach einer Alternative gesucht werden.

Die Normalkraft errechnet sich schließlich, wenn die lokale Elementsteifigkeitsmatrix mit der Transformationsmatrix und den globalen Verschiebungen der Stabknotenpunkte multipliziert werden:

$$N_i = K_i \cdot T_i \cdot u_i$$

1.8 Ergebnisse_Ausgeben()

```
function [] = Ergebnisse_Ausgeben(sDATEINAME,aU_ges,aN_ges)
    %KOPIE_DER_EINLESEDATEI_SPEICHERN---------------------------------------------------------------------------
    fhTEXTDATEI = fopen(sDATEINAME,'r');
    sEINLESEDATEN=fread(fhTEXTDATEI);
    fclose(fhTEXTDATEI);

    %AUSGABETEXTDATEI_ANLEGEN_UND_BESCHREIBEN-------------------------------------------------------------------
    %Initialisierung
    cDATEINAME=char(sDATEINAME);
    while(1)
        cCHAR=cDATEINAME(size(cDATEINAME,2));
        cDATEINAME=cDATEINAME(1:end-1);
        if(cCHAR=='.')
            break;
        end
    end
    sDATEINAME=strcat(cDATEINAME,".erg");
    fhTEXTDATEI = fopen(sDATEINAME,'w+');
    fprintf(fhTEXTDATEI,'%s\n',sEINLESEDATEN);
    sTRENNSTRICH="//----------------------------------------------------------";
    cWERTAUSGABEFORMAT='%.8f';
    fprintf(fhTEXTDATEI,'\n\n\n');
    fprintf(fhTEXTDATEI,'%s\n',sTRENNSTRICH);
    %Verschiebungen
    fprintf(fhTEXTDATEI,'Verschiebungen: \n');
    fprintf(fhTEXTDATEI,'Knoten\t\t\tUx\t\t\t\tUy\n');
    for(i=2:2:size(aU_ges,1))
        cTEXT=strcat('%u\t\t\t\t',cWERTAUSGABEFORMAT,'\t\t',cWERTAUSGABEFORMAT,'\n');
        fprintf(fhTEXTDATEI,cTEXT,i/2,aU_ges(i-1),aU_ges(i));
    end
    fprintf(fhTEXTDATEI,'\n');
    %Stabkräfte
    fprintf(fhTEXTDATEI,'Stabkräfte: \n');
    fprintf(fhTEXTDATEI,'Stab\t\t\tN\n');
    for(i=1:1:size(aN_ges,1))
        cTEXT=strcat('%u\t\t\t\t',cWERTAUSGABEFORMAT,'\n');
        fprintf(fhTEXTDATEI,cTEXT,aN_ges(i,1),aN_ges(i,3));
    end
    fprintf(fhTEXTDATEI,'%s\n',sTRENNSTRICH);

    %SCHLIEßEN_DER_AUSGABETEXTDATEI-----------------------------------------------------------------------------
    fclose(fhTEXTDATEI);
end
```

Zunächst wird der aus dem Einlesevorgang übergebene Dateiname nocheinmal genutzt und die Datei komplett auf einen String eingelesen und danach geschlossen.

Dann wird der Dateiname von der Endung befreit und mit der Endung „.dat“ versehen. Eine neue Datei wird angelegt und der zuvor eingelesene String hineinkopiert.

Unter die Eingangsdaten werden dann die Verschiebungen und Stabkräfte aufgereiht.

1.9 Verformung_Plotten()

```
function [] = Verformung_Plotten(aSTABVERBINDUNGEN,aKNOTENKOORDINATEN,aU_ges,aN_ges,aL,aEA)
    rgbUNVERFORMT=[0.5 0.5 0.5];
    iKNOTENDURCHMESSER=125;
    dSKALIERUNG=(10/max(abs(aU_ges(:))))*max(aL(:));
    dKRAFTMAXIMUM=max(abs(aN_ges(:,2)));
    dSTEIFIGKEITSMAXIMUM=max(aEA(:,2));
    for(i=1:1:size(aSTABVERBINDUNGEN,1))
        dKRAFT_IN_PROZENT=abs(aN_ges(i,2))/dKRAFTMAXIMUM;
        rgbVERFORMT(1:3)=[0+dKRAFT_IN_PROZENT 1-dKRAFT_IN_PROZENT 0];
        dLINIENSTAERKE=(aEA(i,2)/dSTEIFIGKEITSMAXIMUM)*2;
        iANFANGSKNOTEN=aSTABVERBINDUNGEN(i,2);
        iENDKNOTEN=aSTABVERBINDUNGEN(i,3);
        X(i,1:2)=[ ...
                    aKNOTENKOORDINATEN(iANFANGSKNOTEN,2)+(aU_ges((2*iANFANGSKNOTEN)-1)*dSKALIERUNG), ...
                    aKNOTENKOORDINATEN(iENDKNOTEN,2)+(aU_ges((2*iENDKNOTEN)-1)*dSKALIERUNG) ...
                 ];
        Y(i,1:2)=[ ...
                    aKNOTENKOORDINATEN(iANFANGSKNOTEN,3)+(aU_ges((2*iANFANGSKNOTEN))*dSKALIERUNG), ...
                    aKNOTENKOORDINATEN(iENDKNOTEN,3)+(aU_ges(2*iENDKNOTEN)*dSKALIERUNG) ...
                 ];
        plot( ...
                X(i,1:2),Y(i,1:2), ...
                'color',rgbVERFORMT, ...
                'LineWidth', dLINIENSTAERKE ...
            );
        hold on;
        scatter( ...
                X(i,1:2),Y(i,1:2), ...
                iKNOTENDURCHMESSER,'o', 'filled', ...
                'MarkerEdgeColor',rgbVERFORMT,...
                'MarkerFaceColor',[1 1 1],...
                'LineWidth',1.5 ...
               );
        hold on;
    end
    for(i=1:1:size(aSTABVERBINDUNGEN,1))
        iANFANGSKNOTEN=aSTABVERBINDUNGEN(i,2);
        iENDKNOTEN=aSTABVERBINDUNGEN(i,3);
        X(i,1:2)=[aKNOTENKOORDINATEN(iANFANGSKNOTEN,2),aKNOTENKOORDINATEN(iENDKNOTEN,2)];
        Y(i,1:2)=[aKNOTENKOORDINATEN(iANFANGSKNOTEN,3),aKNOTENKOORDINATEN(iENDKNOTEN,3)];
        plot( ...
                X(i,1:2),Y(i,1:2), ...
                'color',rgbUNVERFORMT, ...
                'LineWidth',dLINIENSTAERKE ...
            );
        hold on;
        scatter( ...
                X(i,1:2),Y(i,1:2), ...
                iKNOTENDURCHMESSER,'o', 'filled', ...
                'MarkerEdgeColor',rgbUNVERFORMT,...
                'MarkerFaceColor',[1 1 1],...
                'LineWidth',1.5 ...
               );
        hold on;
        text( ...
                X(i,1)-(iKNOTENDURCHMESSER/2),Y(i,1), ...
                cellstr(string(iANFANGSKNOTEN)), ...
                'color',rgbUNVERFORMT ...
             );
        hold on;
        text( ...
                X(i,2)-(iKNOTENDURCHMESSER/2),Y(i,2), ...
                cellstr(string(iENDKNOTEN)), ...
                'color',rgbUNVERFORMT ...
             );
        hold on;
    end
    axis('equal');                  %Seitenverhältnis 1:1
    title('Verschiebung');          %Überschrift
    xlabel('x');                    %Achsenbeschriftung x
    ylabel('y');                    %Achsenbeschriftung y
end
```

Die Darstellung der Stäbe erfolgt über 2D-Linienplot. Die Knoten werde diesen mit Punktplot (gefüllten Kreise) überlagert. Die Dicke der Stäbe wird über die Steifigkeitsnormierung umgesetzt und mit einem konstanten Faktor multipliziert:

$$t_i = Faktor \cdot \frac{E_i \cdot A_i}{\max\limits_{1 \le i \le i_{max}} (E_i \cdot A_i)} = \{0, \dots, Faktor\}$$

Das unverformte Fachwerk wird grau $rgb_{unverformt} = \{\,0{,}5;\ 0{,}5;\ 0{,}5\}$ eingefärbt. Für das verformte Fachwerk wird die Stabkraft normiert:

$$F_{i,normiert} = \frac{F_i}{\max\limits_{1 \le i \le i_{max}} (F_i)} = \{0, \dots, 1\}$$

Die Belastung der Stäbe wird dann mit den Farben rot und grün dargestellt:

$$rgb_{verformt,i} = \left\{\, 0 + F_{i,normiert};\ 1 - F_{i,normiert};\ 0 \right\}$$

Für die verformte Darstellung wird zunächst die Verschiebung mit einem Faktor $f_{Skalierung}$ vergrößert, um sie besser sichtbar zu machen. Der Faktor $f_{Skalierung}$ entsteht durch das Verhältnis der größten Verformung zur größten Länge, multipliziert mit einer Konstante:

$$f_{Skalierung} = Faktor \cdot \frac{\max\limits_{1 \le i \le i_{max}} |u_i|}{\max\limits_{1 \le i \le i_{max}} (L_i)}$$

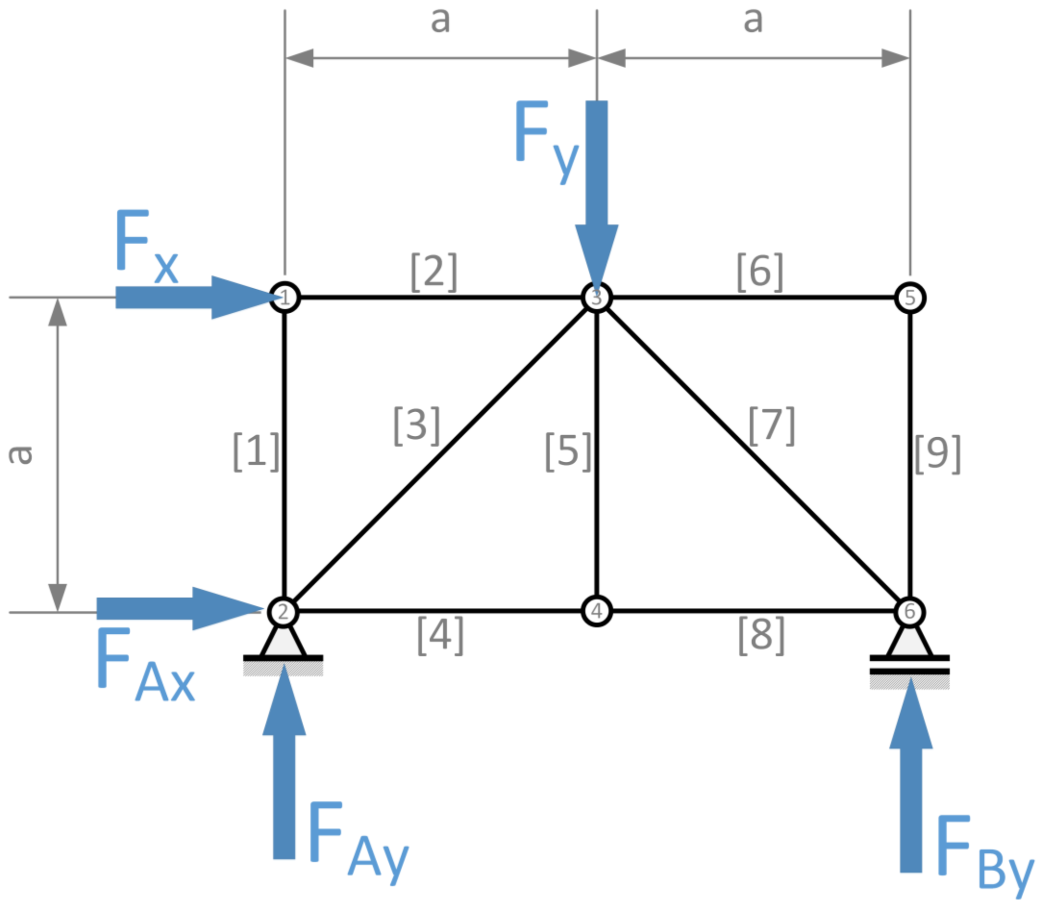
Den ursprünglichen x,y-Koordinaten des Fachwerks werden nun die verlängerten Verschiebungen aufaddiert.

$$XY_{verformt,i} = \begin{pmatrix} X_{unverformt,i} & + & f_{Skalierung} \cdot u_{x,i} \\ Y_{unverformt}, i & + & f_{Skalierung} \cdot u_{y,i} \end{pmatrix}$$

Ostfalia Hochschule für angewandte Wissenschaften
Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476

2 Verifikation

2.1 Auflagerreaktionen und Stabkräfte

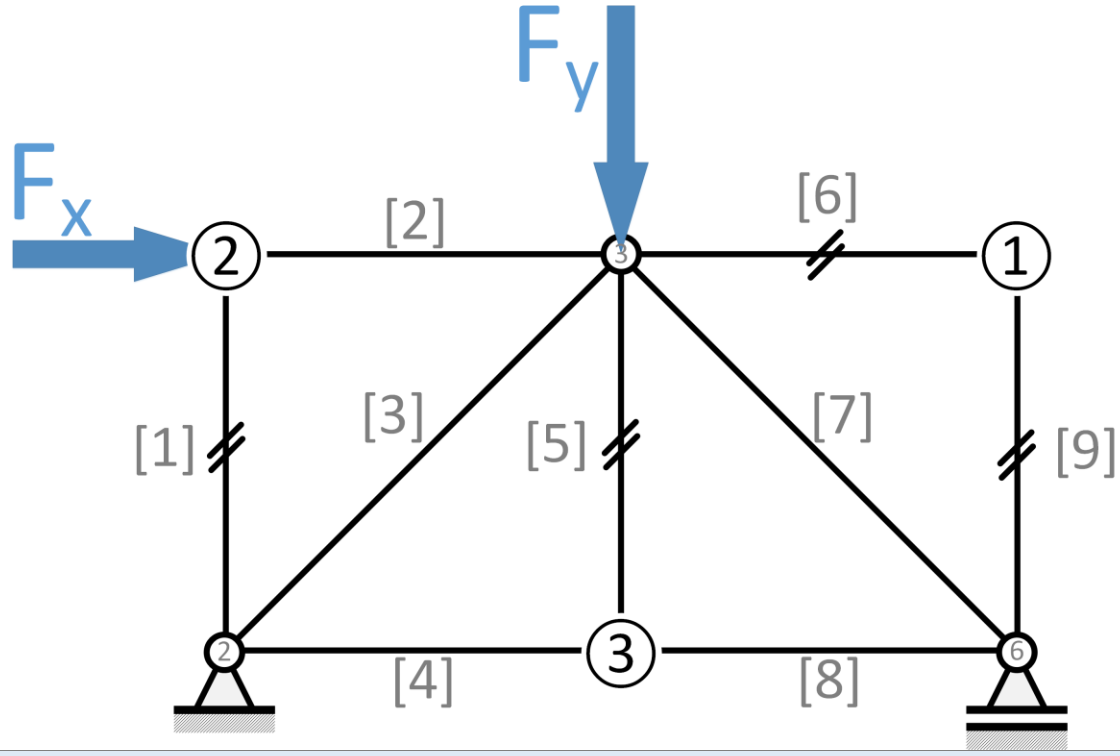


Berechnung der Auflagerreaktionen		
Formeln		Quantitative Größen
		$F_{Ax} = 50000N$ $F_{Ay} = 100000N$ $a = 4m$
$\sum M_{iA} = 0 = -(a \cdot F_{Ax}) - (a \cdot F_{Ay}) + (2 \cdot a \cdot F_{By})$		$F_{By} = \dfrac{50000N + 100000N}{2}$
$\Rightarrow F_{By} = \dfrac{(a \cdot F_{Ax}) + (a \cdot F_{Ay})}{2 \cdot a}$		$\Rightarrow F_{By} = 75000N$
$\sum F_{iy} = 0 = F_{Ay} + F_{By} - F_y$	$\Rightarrow$	$F_{Ay} = 100000N - 75000N$
$\Rightarrow F_{Ay} = F_y - F_{By}$		$\Rightarrow F_{Ay} = 25000N$
$\sum F_{ix} = 0 = F_{Ax} + F_x$		$F_{Ax} = -50000N$
$\Rightarrow F_{Ax} = -F_x$		

Ostfalia Hochschule für angewandte Wissenschaften

Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476

Ermittlung der Nullstäbe
(Quelle: Technische Mechanik 1 | Statik (Gross, Hauger, Schröder, Wall), 14.Auflage S. 151)



① Sind an einem unbelasteten Knoten zwei Stäbe angeschlossen, die nicht in gleicher Richtung liegen („unbelasteter Zweischlag“), so sind beide Stäbe Null-stäbe […].

- Stab [6] ist ein Nullstab $S_6 = 0N$
- Stab [9] ist ein Nullstab $S_9 = 0N$

② Sind an einem belasteten Knoten zwei Stäbe angeschlossen und greift die äußere Kraft in Richtung des einen Stabes an, so ist der andere Stab ein Nullstab […].

- Stab [1] ist ein Nullstab $S_1 = 0N$

③ Sind an einem unbelasteten Knoten drei Stäbe angeschlossen, von denen zwei in gleicher Richtung liegen, so ist der dritte Stab ein Nullstab […].

- Stab [5] ist ein Nullstab $S_5 = 0N$

Stab [1]	$S_1 = 0N$	Nullstab
Stab [2]	$S_2 = -50000N$	Druckstab
Stab [3]	$S_3 = -35355N$	Druckstab
Stab [4]	$S_4 = 75000N$	Zugstab
Stab [5]	$S_5 = 0N$	Nullstab
Stab [6]	$S_6 = 0N$	Nullstab
Stab [7]	$S_7 = -106066N$	Druckstab
Stab [8]	$S_8 = 75000N$	Zugstab
Stab [9]	$S_9 = 0N$	Nullstab

Hinweis

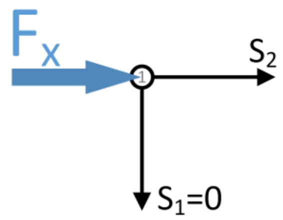
$$\propto = \tan^{-1}\left(\frac{a}{a}\right) = 45°$$

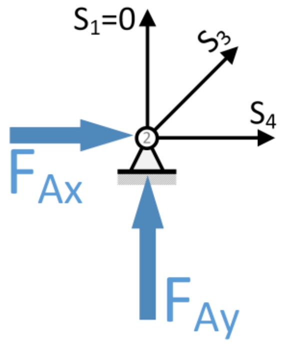
$$\sin(45°) = \cos(45°) = \frac{\sqrt[2]{2}}{2}$$

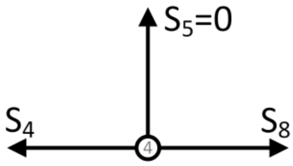
$$\frac{\sqrt[2]{2}}{2} = \frac{\sqrt[2]{2}}{2} \cdot \frac{\sqrt[2]{2}}{\sqrt[2]{2}} = \frac{1}{\sqrt[2]{2}}$$

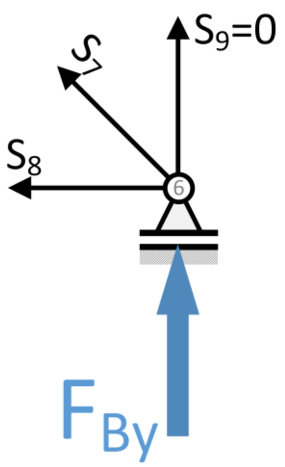
Ostfalia
Hochschule für angewandte Wissenschaften

Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476

Knoten 1		
	$\sum F_{iy} = 0 = S_1$ $\sum F_{ix} = 0 = F_x + S_2$ $\Rightarrow S_2 = -F_x$	$\Rightarrow S_2 = -50000N$

Knoten 2		
	$\sum F_{iy} = 0 = F_{Ay} + S_1 + \frac{S_3}{\sqrt[2]{2}}$ $\Rightarrow S_3 = \left(-F_{Ay} - S_1\right) \cdot \sqrt[2]{2}$ $\sum F_{ix} = 0 = F_{Ax} + S_4 + \frac{S_3}{\sqrt[2]{2}}$ $\Rightarrow S_4 = -\frac{S_3}{\sqrt[2]{2}} - F_{Ax}$	$\Rightarrow S_3 = (-25000N - 0N) \cdot \sqrt[2]{2}$ $\Rightarrow S_3 = -35355N$ $\Rightarrow S_4 = - - \frac{35355N}{\sqrt[2]{2}} - -50000N$ $\Rightarrow S_4 = 75000N$

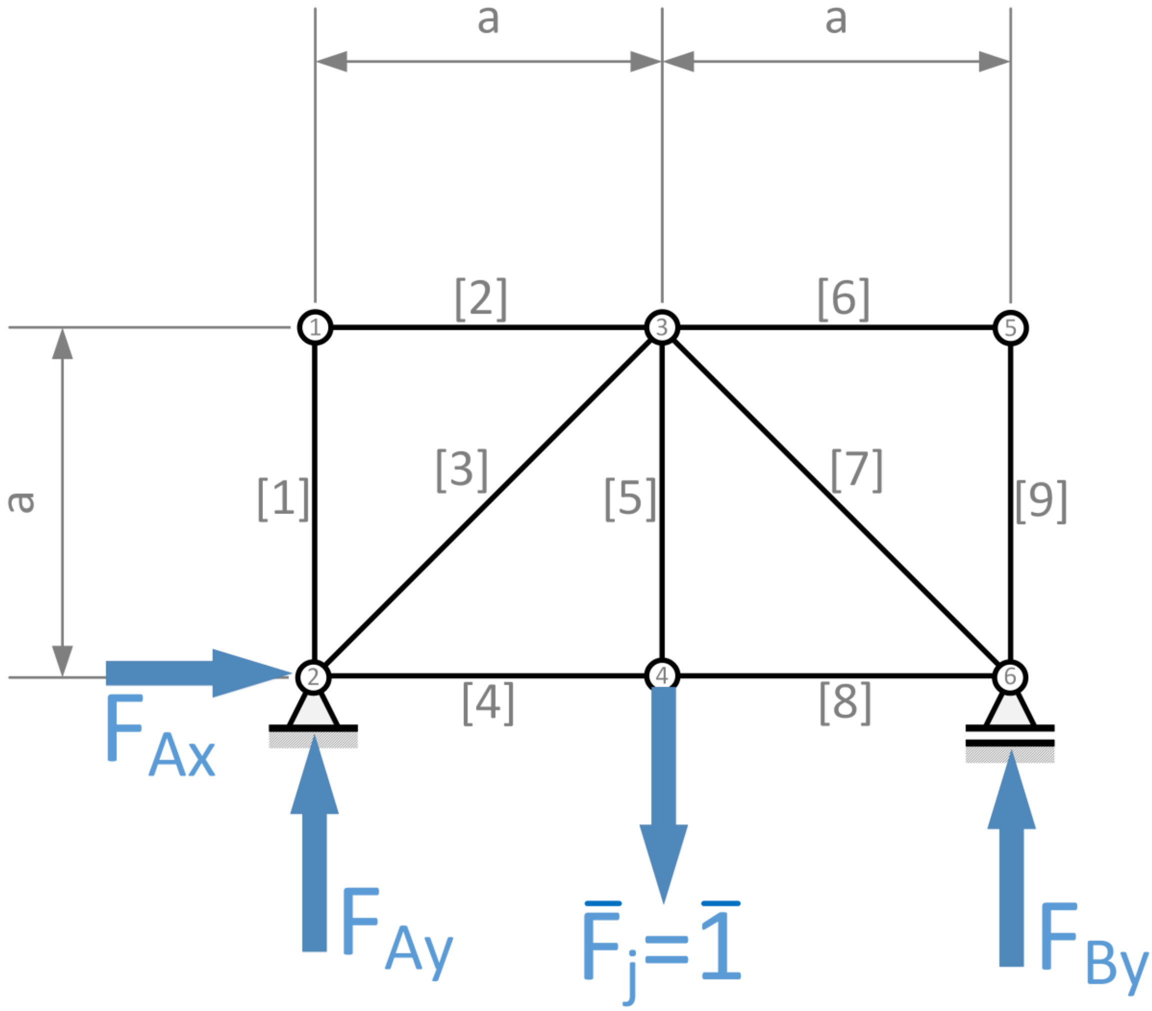
Knoten 4		
	$\sum F_{iy} = 0 = S_5$ $\sum F_{ix} = 0 = -S_4 + S_8$ $\Rightarrow S_8 = S_4$	$\Rightarrow S_8 = 75000N$

Knoten 6		
	$\sum F_{iy} = 0 = S_9 + \frac{S_7}{\sqrt[2]{2}} + F_{Bx}$ $\sum F_{ix} = 0 = -S_8 - \frac{S_7}{\sqrt[2]{2}}$ $\Rightarrow S_7 = \sqrt[2]{2} \cdot -S_8$	$\Rightarrow S_7 = -106066N$

Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476

2.2 Verformung in y am Knoten 4 mit dem Prinzip der Virtuellen Kräfte (PvK)

Zur Bestimmung der vertikalen Verformung wird an dem Punkt j die Virtuelle Kraft F_j eingebracht, zu $F_j = \bar{1}$.



Virtuelle Auflagerreaktionen		
$\sum F_{ix} = 0 = F_{Ax}$ $\Rightarrow F_{Ax} = 0$	$\sum M_{iA} = 0 = (2 \cdot a \cdot F_{By}) - (\bar{1} \cdot a)$ $\Rightarrow F_{By} = \frac{\bar{1} \cdot a}{2 \cdot a}$ $\Rightarrow F_{By} = \frac{\bar{1}}{2}$	$\sum F_{iy} = 0 = F_{Ay} + F_{By} - \bar{1}$ $\Rightarrow F_{Ay} = \bar{1} - F_{By} = \bar{1} - \frac{\bar{1}}{2} = \frac{\bar{1}}{2}$ $\Rightarrow F_{Ay} = \frac{\bar{1}}{2}$

Ostfalia
Hochschule für angewandte Wissenschaften

Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476

Stabkräfte mit virtueller F_j	
Knoten 1	
$\sum F_{ix} = 0 = F_{Ax} + S_4 + \frac{S_3}{\sqrt[2]{2}}$	
$\sum F_{iy} = 0 = F_{Ay} + S_1 + \frac{S_3}{\sqrt[2]{2}}$	
Knoten 2	
$\sum F_{ix} = 0 = S_2$	$\Rightarrow S_2 = 0$
$\sum F_{iy} = 0 = -S_1$	$\Rightarrow S_1 = 0$
Knoten 3	
$\sum F_{ix} = 0 = -S_4 + S_8$	$\Rightarrow S_4 = S_8 = \frac{\bar{1}}{2}$
$\sum F_{iy} = 0 = S_5 - \bar{1}$	$\Rightarrow S_5 = \bar{1}$
Knoten 4	
$\sum F_{ix} = 0 = -S_2 + S_6 - \frac{S_3}{\sqrt[2]{2}} + \frac{S_7}{\sqrt[2]{2}}$	$\Rightarrow S_3 = \frac{S_7}{\sqrt[2]{2}} \cdot \sqrt[2]{2} = S_7 = -\frac{\bar{1}}{2} \cdot \sqrt[2]{2}$
$\sum F_{iy} = 0 = -S_5 - \frac{S_3}{\sqrt[2]{2}} - \frac{S_7}{\sqrt[2]{2}}$	$\rightarrow alles\ bekannt$
Knoten 5	
$\sum F_{ix} = 0 = -\frac{S_7}{\sqrt[2]{2}} - S_8$	$\Rightarrow S_8 = -\frac{S_7}{\sqrt[2]{2}} = -\frac{1}{\sqrt[2]{2}} \cdot -\frac{\bar{1}}{2} \cdot \sqrt[2]{2} = \frac{\bar{1}}{2}$
$\sum F_{iy} = 0 = S_9 + \frac{S_7}{\sqrt[2]{2}} + F_{By}$	$\Rightarrow S_7 = -F_{By} \cdot \sqrt[2]{2} = -\frac{\bar{1}}{2} \cdot \sqrt[2]{2}$
Knoten 6	
$\sum F_{ix} = 0 = -S_6$	$\Rightarrow S_6 = 0$
$\sum F_{iy} = 0 = -S_4$	$\Rightarrow S_9 = 0$

$\Downarrow$

Auflistung der virtuellen Stabkräfte

S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9
0	0	$-\frac{\bar{1}}{2} \cdot \sqrt[2]{2}$	$\frac{\bar{1}}{2}$	$\bar{1}$	0	$-\frac{\bar{1}}{2} \cdot \sqrt[2]{2}$	$\frac{\bar{1}}{2}$	0

Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476

Arbeitssatz PvK:
$$\bar{1} \cdot f_{ik} = \int_0^l \left(\frac{\bar{N}_i \cdot N_k}{E \cdot A}\right) dx + \int_0^l \left(\frac{\bar{M}_i \cdot M_k}{E \cdot A}\right) dx + \int_0^l \left(\frac{\bar{T}_i \cdot T_k}{E \cdot A}\right) dx$$
Da keine Moment- oder Torsionskomponenten anliegen fallen das Integral 2 & 3 weg. Die Normalkraft im Fachwerk ergibt sich aus der/den Stabkräften, somit wird $N \longmapsto S$. Die Verschiebungen f werden stets in Richtungen der Stäbe angegeben $f \longmapsto u$.
$$\bar{1} \cdot u = \int_0^l \left(\frac{\bar{S}_i \cdot S_k}{E \cdot A}\right) dx$$ $$\Rightarrow \bar{1} \cdot u = \frac{1}{E \cdot A} \cdot \int_0^l (\bar{N}_i \cdot N_k) dx$$

Die Ergebnisse der in Aufgabe 2.1 errechnenten Stabkräfte S_k und die zuvor errechneten virtuellen Stabkräfte $\bar{S}_i$ sind in folgender Tabelle aufgeführt:

j	S_k	$\bar{S}_i$	$\bar{S}_i \cdot S_k$	l
1	0	0	0	a
2	$-F_x$	0	0	a
3	$\left(-\frac{F_y}{2} + \frac{F_x}{2}\right) \cdot \sqrt[2]{2}$	$-\frac{\bar{1}}{2} \cdot \sqrt[2]{2}$	$(F_y - F_x) \cdot \frac{\bar{1}}{2}$	$\sqrt[2]{2} \cdot a$
4	$\frac{F_y}{2} + \frac{F_x}{2}$	$\frac{\bar{1}}{2}$	$(F_y + F_x) \cdot \frac{\bar{1}}{4}$	a
5	0	$\bar{1}$	0	a
6	0	0	0	a
7	$\left(-\frac{F_y}{2} + \frac{F_x}{2}\right) \cdot \sqrt[2]{2}$	$-\frac{\bar{1}}{2} \cdot \sqrt[2]{2}$	$(F_y - F_x) \cdot \frac{\bar{1}}{2}$	$\sqrt[2]{2} \cdot a$
8	$\frac{F_y}{2} + \frac{F_x}{2}$	$\frac{\bar{1}}{2}$	$(F_y + F_x) \cdot \frac{\bar{1}}{4}$	a
9	0	0	0	a

Das ergibt für :

$$j = 1 \qquad 0kN \cdot 5m = \qquad 0kNm$$

$$j = 2 \qquad 0kN \cdot 5m = \qquad 0kNm$$

$$j = 3 \qquad (100kN - 50kN) \cdot \frac{\overline{1}}{2} \cdot \sqrt[2]{2} \cdot 5m = \qquad 176{,}7766953kNm$$

$$j = 4 \qquad (100kN + 50kN) \cdot \frac{\overline{1}}{4} \cdot 5m = \qquad 187{,}5000000kNm$$

$$j = 5 \qquad 0kN \cdot 5m = \qquad 0kNm$$

$$j = 6 \qquad 0kN \cdot 5m = \qquad 0kNm$$

$$j = 7 \qquad (100kN - 50kN) \cdot \frac{\overline{1}}{2} \cdot \sqrt[2]{2} \cdot 5m = \qquad 530{,}3300859kNm$$

$$j = 8 \qquad (100kN + 50kN) \cdot \frac{\overline{1}}{4} \cdot 5m = \qquad 187{,}5000000kNm$$

$$j = 9 \qquad 0kN \cdot 5m = \qquad 0kNm$$

$$\sum \int_0^l (\overline{N}_i \cdot N_k) dx = \qquad 1082{,}106781kNm$$

Die Dehnsteifigkeit ist für alle Rundstäbe gleich und ergibt sich aus dem gegebenen E-Modul und dem angegebenen Durchmesser:

$$E = 210000 \frac{N}{mm^2} \qquad E \cdot A = E \cdot \frac{\pi \cdot d^2}{4} \qquad \Rightarrow E \cdot A = 210000 \frac{N}{mm^2} \cdot \frac{\pi \cdot 10^2 \cdot mm^2}{4}$$

$$d = 10mm \qquad \Rightarrow E \cdot A = 16443361{,}43N$$

Damit kann nun die Verschiebung errechnet werden:

$$\overline{1} \cdot u = \frac{1}{E \cdot A} \cdot \int_0^l (\overline{N}_i \cdot N_k) dx$$

$$\Rightarrow u = \frac{1}{16443361{,}43N} \cdot \frac{1082{,}106781kNm}{\overline{1}}$$

$$\Rightarrow u = 0{,}065608m = 65{,}608mm$$

Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476

2.3 FEM-Programm Knotenverformungen und Stabkräfte

Normalkräfte	
Stab	$N\ [N]$
1	0.00000000
2	−50000.00000000
3	−35355.33905933
4	75000.00000000
5	0.00000000
6	0.00000000
7	−106066.01717798
8	75000.00000000
9	0.00000000

Verschiebungen		
Knoten	$u_x\ [mm]$	$u_y\ [mm]$
1	59.36022994	0.00000000
2	0.00000000	0.00000000
3	44.19492815	−65.64190362
4	22.74795268	−65.64190362
5	44.19492815	0.00000000
6	45.49590537	0.00000000

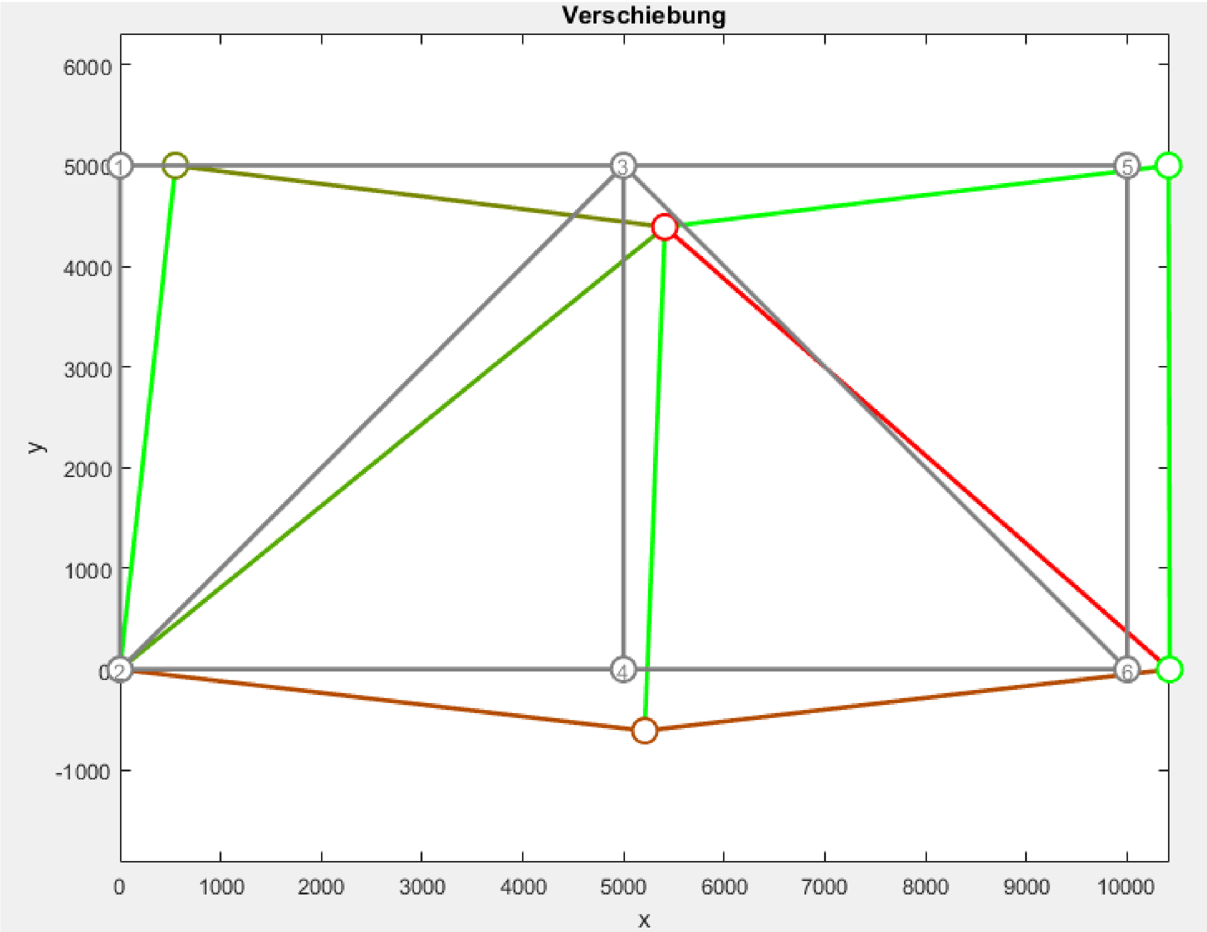


Abbildung 2.1: Antwort des FEM-Programms mit ≈100-fach übertriebener Verschiebung

Thorsten Wollny	70316173
Alexander König	70012769
Florian Franke	70406476

2.4 Vergleich Handrechnung mit FEM. Warum müssen sie exakt übereinstimmen?

Vergleich der Stabkräfte:

Aufgabe	Normalkraft in Stab [kN]								
	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9
$A\ 2.1$	$0{,}0000$	$-50{,}0000$	$-35{,}3553$	$75{,}0000$	$0{,}0000$	$0{,}0000$	$-106{,}066$	$75{,}0000$	$0{,}0000$
$A\ 2.3$	$0{,}0000$	$-50{,}0000$	$-35{,}3553$	$75{,}0000$	$0{,}0000$	$0{,}0000$	$-106{,}066$	$75{,}0000$	$0{,}0000$

Vergleich der vertikalen Verschiebung

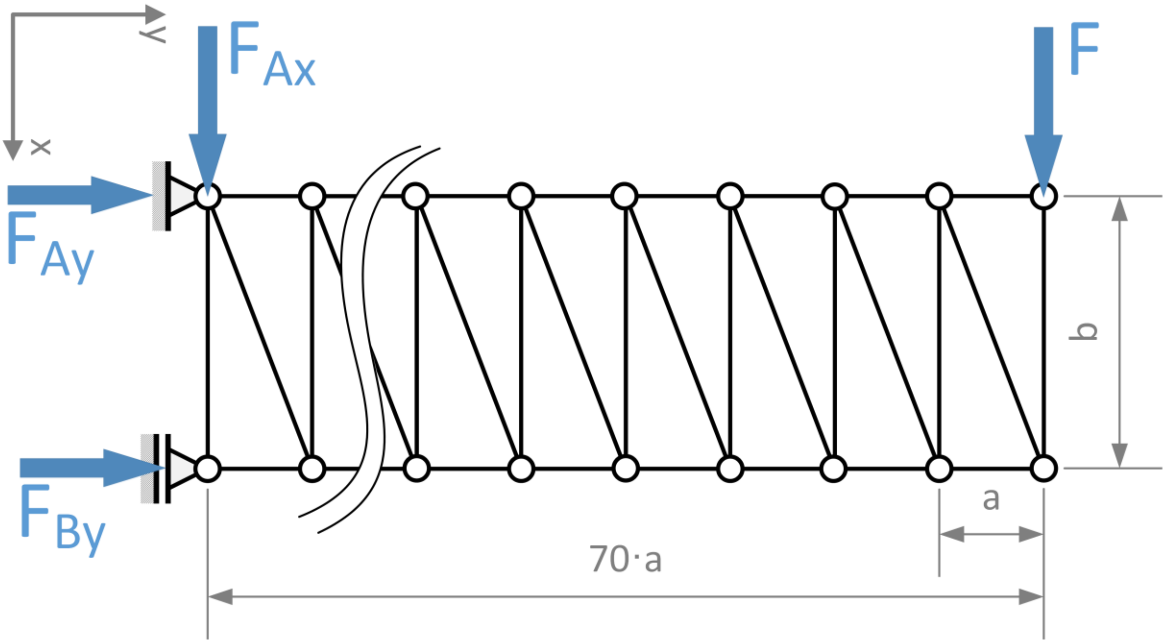
Aufgabe	In Punkt j (Knoten K3/K4)
	$uy\ [mm]$
$A\ 2.2$	$65{,}60862597$
$A\ 2.3$	$-65{,}64190362$

Der Vorzeichenwechsel basiert auf einer anderen Festlegung, jedoch bewegen sich beide Verschiebungen nach unten! Die Abweichung in der 2ten Nachkommastelle kommen durch Rundungsfehler zu Stande. Sie werden daher als exakt angesehen.

Sie müssen auch exakt übereinstimmen, da hier ein Stabwerk und dessen Stabkräfte berechnet werden, in dem die Stäbe bzw. die Stabkräfte die Finiten Elemente mit hinreichender Größe / Genauigkeit wiederspiegeln.

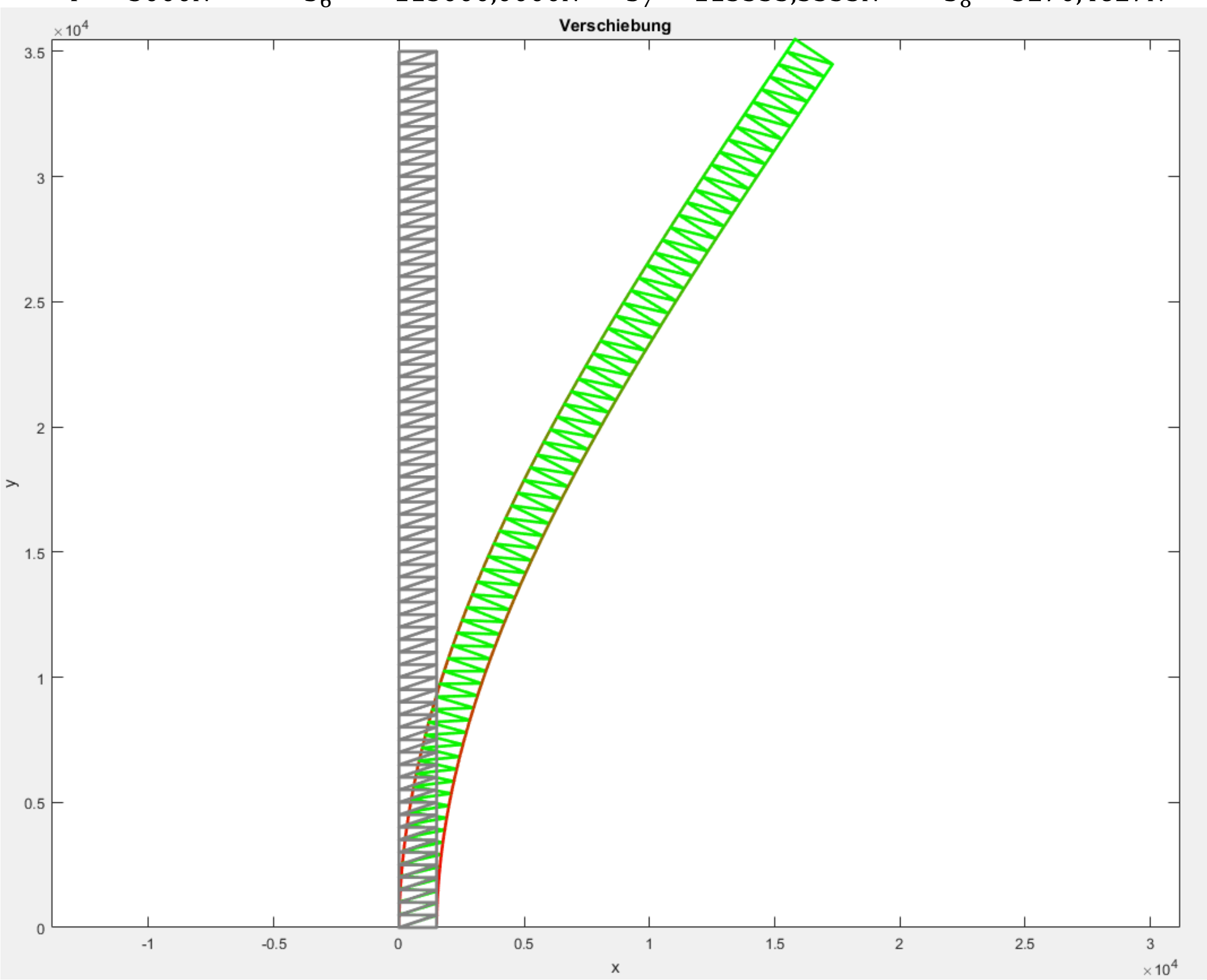
Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476

3 Anwendungsbeispiel

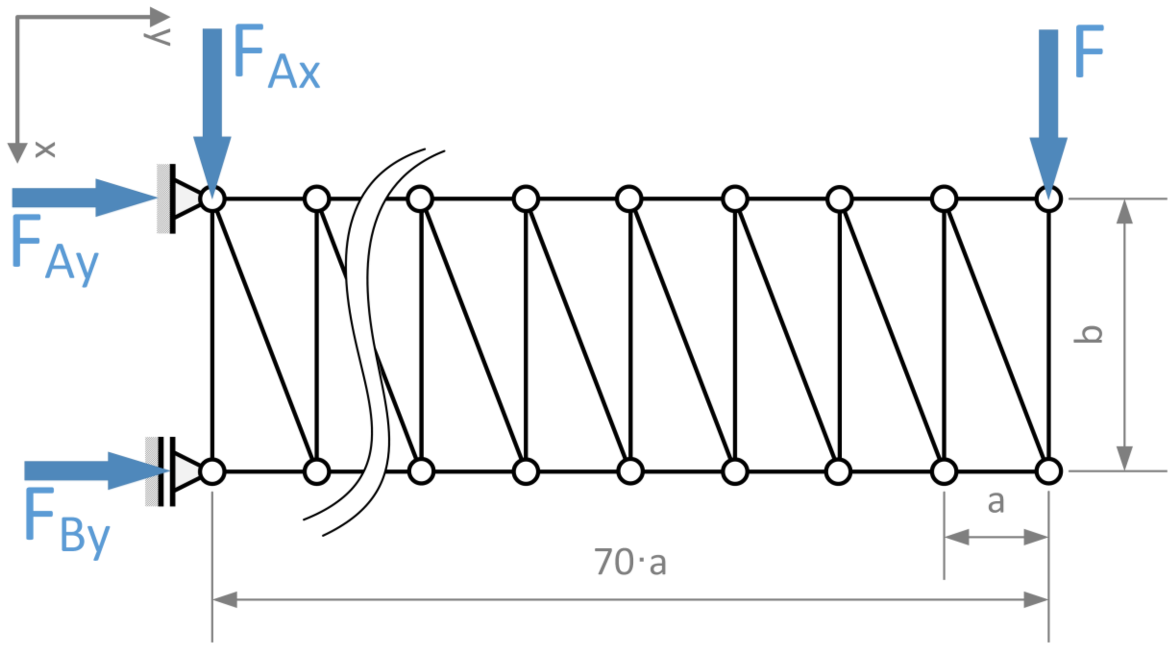


3.1 Verformungen aller Knoten und aller Stab-Kräfte

$F = 5000N$ $S_6 = -115000{,}0000N$ $S_7 = 113333{,}3333N$ $S_8 = 5270{,}4627N$



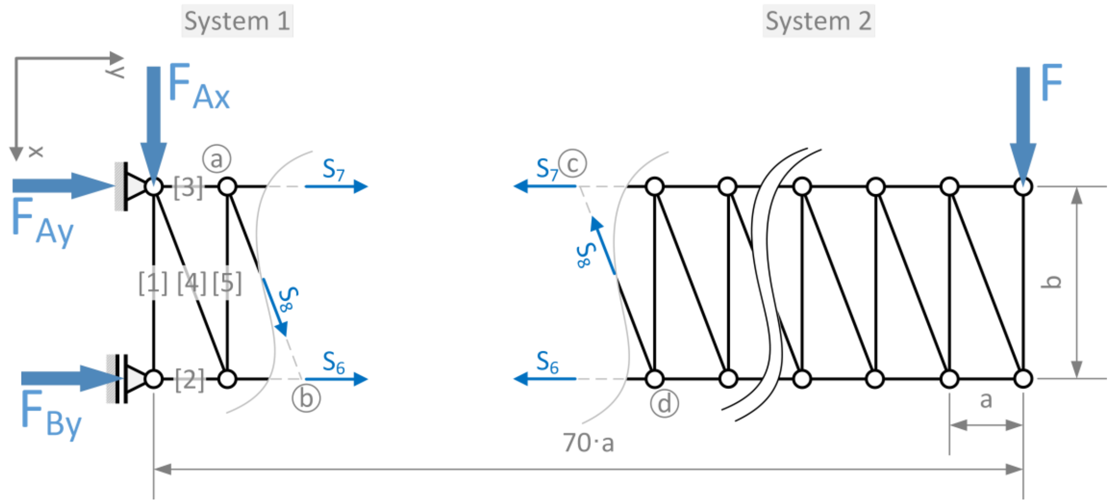
3.2 Verifikation drei beliebiger Stabkräfte mit dem Ritterschnittverfahren



Berechnung der Auflagerreaktionen		
Formeln		Quantitative Größen
		$F = 5000N$ $a = 0{,}5m$ $b = 1{,}5m$
$\sum M_{iA} = 0 = -(70 \cdot a \cdot F) + (b \cdot F_{By})$		$F_{By} = \frac{70 \cdot 0{,}5m \cdot 5000N}{1{,}5m}$
$\Rightarrow F_{By} = \frac{70 \cdot a \cdot F}{b}$		$\Rightarrow F_{By} = 116666{,}\bar{6}N$
$\sum F_{iy} = 0 = F_{Ay} + F_{By}$ $\Rightarrow F_{Ay} = -F_{By}$	$\Rightarrow$	$\Rightarrow F_{Ay} = -116666{,}\bar{6}N$
$\sum F_{ix} = 0 = F_{Ax} + F$ $\Rightarrow F_{Ax} = -F$		$\Rightarrow F_{Ax} = -5000N$

Ostfalia
Hochschule für angewandte Wissenschaften

Thorsten Wollny 70316173
Alexander König 70012769
Florian Franke 70406476



System 1

$$\sum M_{i,a} = 0 = b \cdot S_6 + b \cdot F_{By} + a \cdot F_{Ax}$$

$$\Rightarrow S_6 = -F_{By} - \frac{a \cdot F_{Ax}}{b}$$

$$\Rightarrow S_6 = -116666,\bar{6}N - \frac{0{,}5m \cdot -5000N}{1{,}5m}$$

$$\Rightarrow S_6 = -115000N$$

$$\sum M_{i,b} = 0 = -\, b \cdot F_{Ay} - b \cdot S_7 + (2 \cdot a \cdot F_{Ax})$$

$$\Rightarrow S_7 = -F_{Ay} + \frac{2 \cdot a \cdot F_{Ax}}{b}$$

$$\Rightarrow S_7 = -\,-\,116666,\bar{6}N + \frac{2 \cdot 0{,}5m \cdot -5000N}{1{,}5m}$$

$$\Rightarrow S_7 = 113333,\bar{3}N$$

$$\alpha = \tan^{-1}\left(\frac{a}{b}\right)$$

$$\Rightarrow \alpha = \tan^{-1}\left(\frac{0{,}5m}{1{,}5m}\right) = 18{,}4°$$

$$\sum F_{i,y} = 0 = F_{Ay} + F_{By} + S_6 + S_7 + \sin(\alpha) \cdot S_8$$

$$\Rightarrow S_8 = \frac{-F_{By} - F_{Ay} - S_6 - S_7}{\sin\left(\tan^{-1}\left(\frac{a}{b}\right)\right)}$$

$$\Rightarrow S_8 = -\frac{116667N}{\sin(18{,}4°)} - \frac{-116667N}{\sin(18{,}4°)} - \frac{-115000N}{\sin(18{,}4°)} - \frac{113333N}{\sin(18{,}4°)} = 5271{,}5N$$

System 2

$$\sum M_{i,c} = 0 = -\, b \cdot S_6 - (69 \cdot a \cdot F)$$

$$\Rightarrow S_6 = -\frac{69 \cdot a \cdot F}{b}$$

$$\Rightarrow S_6 = -\frac{69 \cdot 0{,}5m \cdot 5000N}{1{,}5m}$$

$$\Rightarrow S_6 = -115000N$$

$$\sum M_{i,d} = 0 = b \cdot S_7 - (68 \cdot a \cdot F)$$

$$\Rightarrow S_7 = \frac{68 \cdot a \cdot F}{b}$$

$$\Rightarrow S_7 = \frac{68 \cdot 0{,}5m \cdot 5000N}{1{,}5m}$$

$$\Rightarrow S_7 = 113333,\bar{3}N$$

$$\alpha = \tan^{-1}\left(\frac{a}{b}\right)$$

$$\Rightarrow \alpha = \tan^{-1}\left(\frac{0{,}5m}{1{,}5m}\right) = 18{,}4°$$

$$\sum F_{i,y} = 0 = S_6 + S_7 + \sin(\alpha) \cdot S_8$$

$$\Rightarrow S_8 = \frac{-S_6 - S_7}{\sin\left(\tan^{-1}\left(\frac{a}{b}\right)\right)}$$

$$\Rightarrow S_8 = -\frac{-115000N}{\sin(18{,}4°)} - \frac{113333N}{\sin(18{,}4°)} = 5271{,}5N$$