Hemtal 4. Inlämning 6/11-2013

Lösningarna till uppgifterna ska vara väl presenterade och lätta att följa. Det innebär speciellt att införda beteckningar ska definieras, att den logiska strukturen tydligt beskrivs i ord eller symboler och att resonemangen är väl motiverade och tydligt förklarade. Använd gärna Matlab för att kontrollera att du har räknat rätt.

Kom ihåg att skriva lösningarna till varje uppgift på separata blad samt fylla i ett försättsblad. Häfta INTE ihop lösningsbladen. Två av nedanstående uppgifter kommer att samlas in för rättning. Vilka meddelas på lektionen den 6/11.

Uppgift 1

a) Bevisa genom induktion att

$$2^n < n!$$

för varje positivt heltal n, $n \geq 4$.

b) Antag att en linjärt oberoende mängd $S = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_p\}$, spänner ett underrum V till $\mathbb{R}^n$. Visa att om en godtycklig vektor $\mathbf{v}$ i V läggs till S så blir mängden linjärt beroende.

Uppgift 2

Bestäm null(A), row(A) och col(A) för nedanstående matriser. Vad är rank(A)? Verifiera dimensionsteoremet (Sats 7.4.1). I $a) - c)$, utför beräkningarna för hand, i $d) - e)$ använd Matlab.

$$a) \begin{bmatrix} 1 & 2 & 3 \\ 1 & 1 & 2 \\ -1 & 3 & 2 \end{bmatrix}, \quad b) \begin{bmatrix} 1 & 1 & 2 \\ 0 & -2 & 1 \\ 1 & 0 & 2 \end{bmatrix}, \quad c) \begin{bmatrix} 2 & -1 & 1 & -2 \\ -1 & 2 & 1 & 1 \\ 4 & -5 & -1 & -4 \\ -1 & 5 & 4 & 1 \end{bmatrix},$$

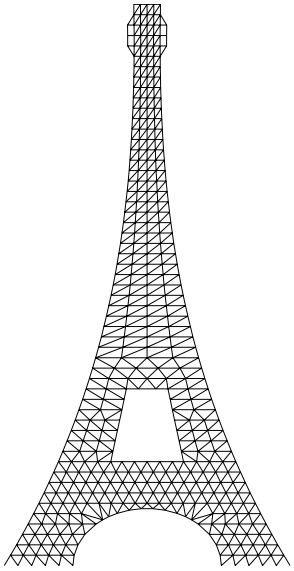
$$d) \begin{bmatrix} 2 & -3 & 1 & 0 & 4 \\ 1 & 1 & 2 & 2 & 0 \\ 3 & 0 & -1 & 4 & 5 \\ 0 & -2 & 4 & -2 & -1 \\ 3 & -4 & 7 & 0 & 3 \end{bmatrix}. \quad e) \begin{bmatrix} 1 & -2 & 1 & 0 \\ 1 & 0 & 4 & 2 \\ 0 & 1 & -2 & -1 \\ 2 & 1 & -3 & 1 \\ 2 & -2 & -1 & 1 \end{bmatrix}.$$

Datorlaboration

I den här delen av hemtalet ska ni göra en datorlabb. Svaren ska redovisas i form av skriftliga svar på frågor som ställs, utskrift av de plottar som efterfrågas samt en kort beskrivning av vad ni har gjort på respektive uppgift.

I många realistiska tillämpningar måste man lösa *stora* linjära ekvationsystem, med miljontals obekanta. Det är i dessa fall som effektiva algoritmer blir viktiga att använda.

Som exempel ska ni här räkna på ett komplicerat fackverk: en modell av Eiffeltornet. Ett fackverk består av stänger förenade genom leder i ett antal noder. Ni ska beräkna deformationen av fackverket när noderna belastas av yttre krafter. Ekvationerna för deformationen härleds i hållfasthetsläran, och baseras på att förskjutningarna i varje nod är små (linjär teori), och att Hookes lag gäller för förlängningen av varje stång. (**Notera att för vissa noder kommer förskjutningarna bli för stora för att den linjära teorin ska gälla och resultatet är inte tillförlitligt.**)



Figur 1: Modellen i `eiffel2.mat`, med 399 noder (798 obekanta).

I slutändan får man ett linjärt ekvationssystem på formen $A\mathbf{x} = \mathbf{b}$. När antalet noder i fackverket är N kommer antalet obekanta vara $2N$ och $A \in \mathbb{R}^{2N\times 2N}$. Matrisen A brukar kallas *styvhetsmatrisen*. Högerledet $\mathbf{b}$ innehåller de givna yttre krafterna som verkar på noderna,

$$\mathbf{b} = (F_1^x, F_1^y, F_2^x, F_2^y, \ldots, F_N^x, F_N^y)^T, \qquad \mathbf{b} \in \mathbb{R}^{2N},$$

där $\mathbf{F}_j = (F_j^x, F_j^y)^T$ är kraften i nod j. Lösningen $\mathbf{x}$ innehåller de resulterande (obekanta) förskjutningarna,

$$\mathbf{x} = (\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2, \ldots, \Delta x_N, \Delta y_N)^T, \qquad \mathbf{x} \in \mathbb{R}^{2N}.$$

Här är alltså $(\Delta x_j, \Delta y_j)^T$ förskjutningen av nod j när fackverket belastas med krafterna i $\mathbf{b}$.

På kurssidan (kurswebben) under fliken **Programbibilotek** finns filerna `eiffel1.mat`, `eiffel2.mat`, `eiffel3.mat` och `eiffel4.mat`. De innehåller fyra olika modeller av Eiffeltornet med växande detaljrikedom (N = 261, 399, 561, 1592). Varje modell består av nodkoordinater i vektorerna `xnod, ynod`, stångindex i matrisen `bars` (används bara för plottningen) och styvhetsmatrisen `A`.

Uppgift 3

Ladda in en av modellerna i Matlab med kommandot `load`. Hämta funktionsfilen `trussplot.m` från programbiblioteket och anropa den med `trussplot(xnod,ynod,bars)` för att plotta tornet. Välj nu en av noderna och belasta den med en kraft rakt högerut med beloppet ett. (Sätt $F_j^x = 1$ för något j, och resten av elementen i $\mathbf{b}$ lika med noll.) Lös systemet $A\mathbf{x} = \mathbf{b}$ med backslash för att få fram förskjutningarna i alla punkter. Beräkna de nya koordinaterna för det belastade tornet, $x_j^{\text{bel}} = x_j + \Delta x_j$, etc.:

```
xbel = xnod + x(1:2:end); ybel = ynod + x(2:2:end);
```

Plotta det belastade tornet. Använd `hold on` för att plotta de två tornen ovanpå varandra i samma figur. Markera vilken nod ni valt.

Skriv ut och redovisa plotten med den markerade noden.

Uppgift 4

Backslash-kommandot i Matlab använder normalt vanlig gausseliminering för att lösa ekvationsystemet. Undersök hur tidsåtgången för gausseliminering beror på systemmatrisens storlek genom att lösa ekvationsystemet $A\mathbf{x} = \mathbf{b}$ med ett godtyckligt valt högerled $\mathbf{b}$ för var och en av de fyra modellerna. Använd Matlab-kommandot `cputime` alternativt Matlab-kommandona `tic` och `toc`. (`help cputime` eller `help tic` ger mer info.) För att få bra noggrannhet i mätningen av tiden (speciellt om den är kort) bör man upprepa beräkningarna några gånger och ta medelvärdet. Plotta tidsåtgången mot antal obekanta N i en `loglog`-plot. Hur ska tidsåtgången bero på N enligt teorin? Stämmer det?

Redovisa i form av svaren på frågorna samt den efterfrågade plotten.

Uppgift 5+6

Ni ska räkna ut i vilka noder fackverket är mest respektive minst känslig för horisontell belastning. Börja med den minsta modellen, `eiffel1.mat`. Tag en nod i taget. Belasta den med samma kraft som i a) ovan och räkna ut resulterande förskjutningar $\mathbf{x}$. Notera storleken på den totala förskjutningen, dvs $||\mathbf{x}||$. Fortsätt med nästa nod, etc. Systematisera beräkningarna med en `for`-slinga i ert Matlab-program och spara storleken på förskjutningen för varje nod. Ta sedan reda på vilken som nod ger störst respektive minst total förskjutning. Plotta tornet med `trussplot` och markera dessa mest och minst känsliga noder i figuren.

Redovisa och motivera svaret på frågan om vilken nod som ger störst respektive minst förskjutning samt skriv ut plotten på tornet, enligt ovan.

Ni kommer att behöva lösa samma stora linjära ekvationssystem med många olika högerled (N stycken). När matrisen är stor, vilket är fallet för de större modellerna, blir detta mycket tidskrävande. Optimera programmet genom att använda LU-faktorisering av A (Matlab-kommandot `lu`). Uppskatta tidsvinsten av detta. Kvantifiera för olika modeller. Kan ni köra även de större modellerna?

Redovisa den uppskattade tidsvinsten och besvara frågan.

Frivillig uppgift

När en matris är gles kan betydligt effektivare metoder än vanlig gausseliminering användas för att lösa ekvationsystemet. Använd `spy(A)` för att studera styvhetsmatrisens struktur. Vad kan man säga om den?

Genom att tala om för Matlab att matrisen är gles kommer bättre metoder automatiskt användas när backslash anropas. Detta kan ni enkelt göra här genom att skriva `A=sparse(A)`. Gå igenom beräkningarna i **Uppgift 5+6** igen. Hur stor tidsvinst gör man i detta fall genom att låta Matlab använda metoder för glesa matriser? Prova med och utan LU-faktorisering.