Stelsels Lineaire Vergelijkingen - Oefening 5

Ruben Van Assche

5 november 2017

1 Software

Doorheen deze opgave heb ik gebruik gemaakt van GSL[1]. Alle functies in deze opgave gebruikt komen dan ook uit de `gsl_linalg` workspace, de `gsl_sf` workspace en natuurlijk de functies voor matrices(`gsl_matrix`) en vector.(`gsl_vector`).

Informatie over het compileren en runnen van het programma kan gevonden worden in `README.md`.

2 Opgave

Het doel van deze opgave was de berekening van het volgende benchmark probleem:

$$\sum_{j=1}^{n} a_{ij}x_j = \binom{n+i-1}{i} \qquad i = 1, \ldots, n$$

$$a_{ij} = \binom{i+j-2}{j-1} \qquad x_j = 1$$

Hierbij moet een schatting van het conditiegetal gemaakt worden voor de matrix A waarbij $n = 3, 6, 9, 12$. Het stelsel moet voor dezelfde n waarden opgelost worden in dubbele precisie($\frac{1}{2}ULP = 2^{-53}$) en voor deze berekening moet het residu en de error vector berekend worden.

3 De matrix A opstellen

De opgave vereist een beperkte kennis van combinatoriek want in feite is matrix A een Pascal matrix:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 6 \end{bmatrix}$$

[1] `https://www.gnu.org/software/gsl/`

Om deze te bereken wordt er gebruik gemaakt van een combinatie:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Deze komen voor het voorschrift van het benchmark probleem en worden d.m.v. volgende code opgelost:

```
double combinate(const unsigned int n,
    const unsigned int k){
    return ( gsl_sf_fact(n) )/(gsl_sf_fact
        (k) * gsl_sf_fact(n - k));
}
```

Deze code implementeert de gegeven wiskundige definitie van een combinatie d.m.v. `gsl_sf_fact` voor het berekenen van faculteiten.

Het opstellen van de matrices A voor verschillende waarden n kan nu makkelijk gebeuren.

3.1 n = 3

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 6 \end{bmatrix}$$

3.2 n = 6

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 3 & 6 & 10 & 15 & 21 \\ 1 & 4 & 10 & 20 & 35 & 56 \\ 1 & 5 & 15 & 35 & 70 & 126 \\ 1 & 6 & 21 & 56 & 126 & 252 \end{bmatrix}$$

3.3 n = 9

$$\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
1 & 3 & 6 & 10 & 15 & 21 & 28 & 36 & 45 \\
1 & 4 & 10 & 20 & 35 & 56 & 84 & 120 & 165 \\
1 & 5 & 15 & 35 & 70 & 126 & 210 & 330 & 495 \\
1 & 6 & 21 & 56 & 126 & 252 & 462 & 792 & 1287 \\
1 & 7 & 28 & 84 & 210 & 462 & 924 & 1716 & 3003 \\
1 & 8 & 36 & 120 & 330 & 792 & 1716 & 3432 & 6435 \\
1 & 9 & 45 & 165 & 495 & 1287 & 3003 & 6435 & 1.287e+04
\end{bmatrix}$$

3.4 n = 12

$$\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\
1 & 3 & 6 & 10 & 15 & 21 & 28 & 36 & 45 & 55 & 66 & 78 \\
1 & 4 & 10 & 20 & 35 & 56 & 84 & 120 & 165 & 220 & 286 & 364 \\
1 & 5 & 15 & 35 & 70 & 126 & 210 & 330 & 495 & 715 & 1001 & 1365 \\
1 & 6 & 21 & 56 & 126 & 252 & 462 & 792 & 1287 & 2002 & 3003 & 4368 \\
1 & 7 & 28 & 84 & 210 & 462 & 924 & 1716 & 3003 & 5005 & 8008 & 1.238e+04 \\
1 & 8 & 36 & 120 & 330 & 792 & 1716 & 3432 & 6435 & 1.144e+04 & 1.945e+04 & 3.182e+04 \\
1 & 9 & 45 & 165 & 495 & 1287 & 3003 & 6435 & 1.287e+04 & 2.431e+04 & 4.376e+04 & 7.558e+04 \\
1 & 10 & 55 & 220 & 715 & 2002 & 5005 & 1.144e+04 & 2.431e+04 & 4.862e+04 & 9.238e+04 & 1.68e+05 \\
1 & 11 & 66 & 286 & 1001 & 3003 & 8008 & 1.945e+04 & 4.376e+04 & 9.238e+04 & 1.848e+05 & 3.527e+05 \\
1 & 12 & 78 & 364 & 1365 & 4368 & 1.238e+04 & 3.182e+04 & 7.558e+04 & 1.68e+05 & 3.527e+05 & 7.054e+05
\end{bmatrix}$$

4 Conditiegetal

Het conditiegetal geeft aan of een matrix(in deze opgave de matrix A) goed of slecht geconditioneerd is. Het meet dus hoe gevoelig het antwoord is aan storingen in de input data en aan afrondingsfouten gemaakt doorheen het oplossen van de matrix. Dit zien we aan volgende definitie:

$$\frac{\|\delta x\|}{\|x\|} < \kappa(A)\frac{\|\delta b\|}{\|b\|}$$

Hierbij is $\frac{\|\delta b\|}{\|b\|}$ de relatieve wijziging in de rechterkant van het stelsel $Ax = b$ en $\frac{\|\delta x\|}{\|x\|}$ de relatieve wijziging in de oplossing van het stelsel. Wat opvalt is dat het conditiegetal $\kappa(A)$ een vergrotingsfactor is voor relatieve wijzigingen aan de rechterkant van het stelsel die vergrotingen van $\kappa(A)$ kan veroorzaken in de oplossing van het stelsel.

Een groot conditiegetal kan er dus voor zorgen dat de fout in de oplossing van het stelsel gigantisch groot wordt!

4.1 Berekening

Voor het berekenen van het conditiegetal wordt er gebruik gemaakt van een singuliere waarden decompositie(in GSL : `gsl_linalg_SV_decomp`). Deze singuliere waarden σ_i met $i = 1, \ldots, n$ worden gebruikt om het conditiegetal te berekenen via volgende methode :

$$\kappa(A) = \frac{max(\sigma_i)}{min(\sigma_i)} \qquad i = 1, \ldots, n$$

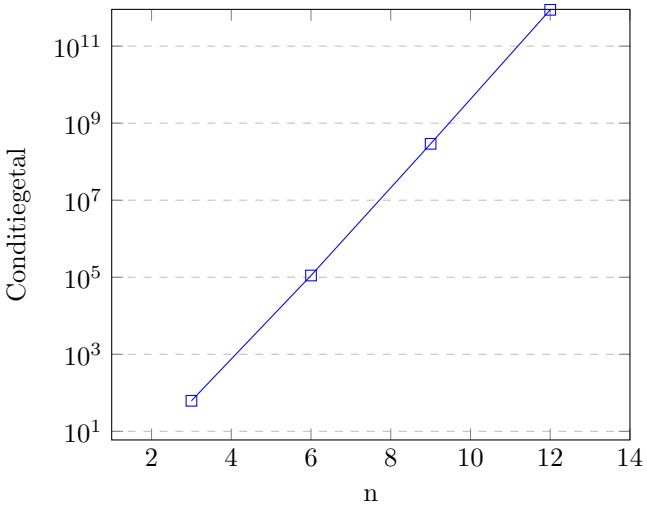
De code voor het berekenen van het conditiegetal is te vinden in `Utils.h`.

4.2 Resultaten

n	Conditiegetal
3	61.98
6	1.108e+05
12	2.908e+08
15	8.764e+11

Wat opvalt is dat naarmate de n groter wordt, het conditiegetal exponentieel groter wordt, dit zal grote gevolgen hebben in latere delen van deze opgave!

Dit zien we als we een grafiek maken van de conditiegetallen(let op er wordt hier gebruik gemaakt van een logaritmische schaal):



Figuur 1: Conditiegetallen voor verschillende n

5 Oplossen stelsel

Voor het oplossen van het stelsel wordt er gebruik gemaakt van GEPP, de functies hiervoor in GSL zijn `gsl_linalg_LU_decomp` en `gsl_linalg_LU_solve`. Hiermee lossen we het stelsel $Ax = b$ met $n = 3, 6, 9, 12$ op.

In de opgave wordt gevraagd om het stelsel in dubbele precisie($\frac{1}{2}ULP = 2^{-53}$) op te lossen. In C/C++ betekent dit het gebruik van het double type. Maar deze kan per machine verschillend zijn. Volgende checkt de precisie van het double type op de machine waar het programma draait:

```
std::cout << "ULP double : " << std::
    numeric_limits<double>::epsilon()
    << std::endl;
std::cout << "ULP required : " << pow
    (2, -52) << std::endl;
```

Wanneer beide functies hetzelfde resultaat opleveren hebben we double precisie zoals gevraagd.

GSL gebruikt voor haar implementatie van matrices en vectoren ook het double type, dit is

zeer belangrijk gezien GEPP geïmplementeerd door GSL gebruik maakt van deze structuren.

5.1 Het Residu en de error vector

Om meer te kunnen zeggen over de correctheid van de oplossing berekenen we voor elke oplossing ook nog het residu:

$$r = y - A\tilde{x}$$

en de error vector :

$$e = x^\star - \tilde{x}$$

Hierbij is $\tilde{x}$ de berekende oplossing en $x^\star$ de exacte wiskundige oplossing.

5.2 Resultaten

5.2.1 n = 3

x	residu	error
-12.01	9.598	13.01
0.8165	5.432	0.1835
0.008774	10.01	0.9912

5.2.2 n = 6

x	residu	error
-544	58.03	545
-22.1	37.07	23.1
2.212	54.47	-1.212
-0.1813	125.9	1.181
0.00149	252	0.9985
-8.086e-07	462	1

5.2.3 n = 9

x	residu	error
-2.843e+04	433.7	2.843e+04
-745.1	207	746.1
-45.54	204.2	46.54
5.018	491.9	-4.018
0.7201	1287	0.2799
0.03267	3003	0.9673
-0.000223	6435	1
2.939e-07	1.287e+04	1
4.364e-11	2.431e+04	1

5.2.4 n = 12

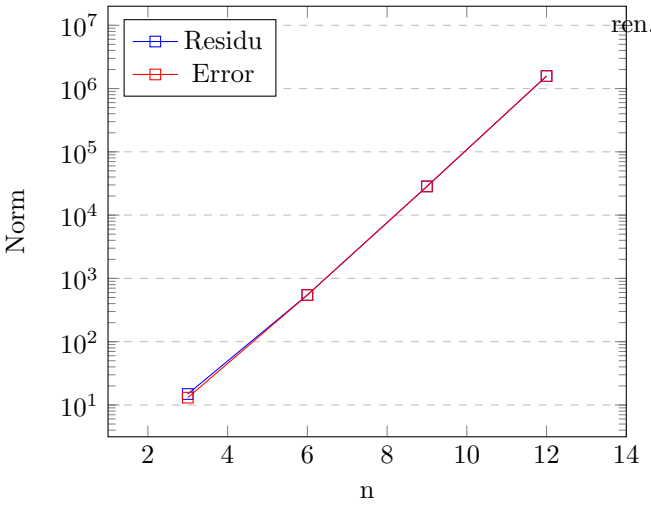
x	residu	error
-1.576e+06	3.051e+04	1.576e+06
-3.05e+04	7621	3.05e+04
-1317	1530	1318
96.84	1319	-95.84
-11.04	4372	12.04
1.795	1.238e+04	-0.795
0.2083	3.182e+04	0.7917
-0.005095	7.558e+04	1.005
3.222e-05	1.68e+05	1
6.252e-08	3.527e+05	1
-2.968e-11	7.054e+05	1
5.3e-12	1.352e+06	1

6 Bespreking Resultaten

Als laatste deel van deze opgave bekijken we de resultaten en de relatie ertussen. Eerst kijken we naar het conditiegetal, zoals gezien op de plot in 4.2 zien we dat naarmate n groter wordt het conditiegetal exponentieel groter wordt. Dit heeft een effect op de residu en error vectoren: de waarden in deze vectoren worden groter.
Om dit te staven heb ik voor elke residu en error vector de euclidische norm berekend d.m.v. `gsl_blas_dnrm2`:

n	residu	error
3	14.89	13.05
6	548.2	545.5
9	2.845e+04	2.844e+04
12	1.577e+06	1.576e+06

Wanneer we deze waarden plotten op een grafiek bekomen we een gelijkaardige grafiek als in 4.2:



ren.

Dit is logisch! Zoals al eerder aangehaald zal naarmate het conditiegetal groter is de fout in de oplossing groter worden. Kijken we naar de x vector voor elke n dan zien we dat deze zelfs al bij een kleine $n = 3$ een zeer grote afwijking van de wiskundig berekende oplossing waarbij elke waarde in de vector 1 is. Dit is dan ook de waarde die we zien in de error vector. Deze geeft aan in hoeverre de berekende oplossing afwijkt van de wiskundige oplossing.

De grootte van het residu volgt naarmate dat n groter wordt de grootte van de error vector. Dit komt omdat het residu weergeeft in hoeverre de oplossing de lineaire condities in $Ax = b$ goed volgt. Gezien de waarden in de x vector voor elke n zo goed als niet in de buurt van 1 liggen, zal bij het oplossen van het stelsel Ax nooit een vector verkregen worden die in de buurt ligt van vector b. Met als gevolg dat het residu ook zeer groot wordt.

6.1 Samengevat

Er is een zeer duidelijke relatie is tussen al deze grootheden berekend in de opgave. Een slecht geconditioneerde matrix A zal zorgen voor een groot conditiegetal, wanneer dan het stelsel $Ax = b$ wordt opgelost zal door het grote conditiegetal de vector x niet in de buurt komen van haar wiskundig berekende versie. Dit zorgt er dan weer voor dat de error vector zeer grote waarden bevat. De waarden in de residu vector zullen dan ook groot worden omdat bij het oplossen van het stelsel Ax met de berekende vector x de uitkomst nooit b zal benade-