Optimization - Assignment 1: Ice on Mars

Anna Frigge
Csongor Horváth

Introduction - Ice Sheet Modeling

- Newtonian fluid which is governed by the full Stokes equations
- Numerical difficulties in solving the full Stokes equations and a commonly used approach is the Shallow Ice Approximation (SIA)

$$a(x) = -\frac{2A}{n+2}(\rho g)^n H(x)^{n+2}\left|\frac{dh}{dx}\right|^{n-1}\frac{dh}{dx}$$

H(x)	- thickness
a(x)	- mass balance function on the surface
ρ	- density of ice
g	- gravitational acceleration
A	- rate factor (~ temperature, between 1e-27 and 1e-23)
n	- flow law exponent (indicates the stiffness of the ice, size commonly in [1, 4])

- From the given measured data we want to use optimization tools to find out the most likely parameter for the unknown data (1 or 2 parameters), by minimizing a given error function

$$f = ||H(x) - H_{obs}(x)||_2^2$$

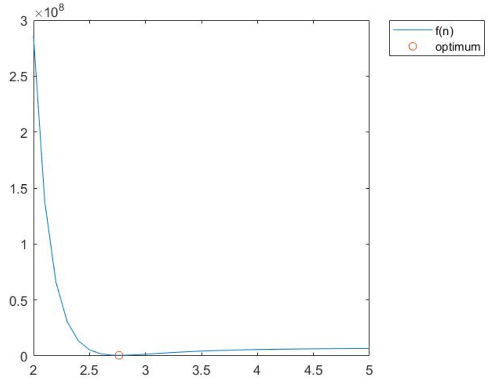
→ least squares fitting with the given data and the known function

UPPSALA
UNIVERSITET

Task 1

- Define function based on exercise with the given data
- Run optimization (fminunc) with or without options
- Choose different starting values in the suggested range
- Plot our result and the function locally

Starting value	Find optimum
2	2.7633
4	2.7633
1 (without options - tol 1e-6)	2.6427 (stopped due to opt tolerance)
1 (with tol 1e-10)	2.7633




```
%% Task 1 %%
options =
optimoptions(@fminunc, 'Display','iter','Algorithm','quasi-newt
on','OptimalityTolerance' ,1e-10);
[n_opt,f1]=fminunc(@f_n,1,options)
%Plot
x=2:0.1:5;
fx=arrayfun(@(x) f_n(x), x);
plot(x,fx, n_opt,f_n(n_opt),  "o")
legend("f(n)", "optimum")
```

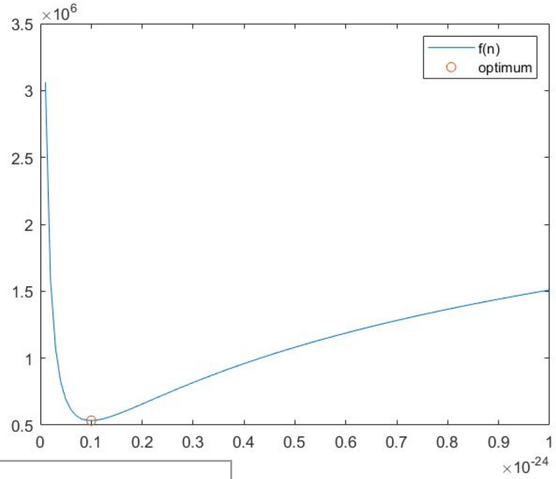
```
function  f = f_n(n)
load('Sharad.mat');
A=10^(-25);
c=-(2*A/(n+2))*(rho*g)^n;
s=abs(dhdx).^(n-1).*dhdx;
a_guess=(c*s);
H=(a./a_guess).^(1/(n+2));
f=norm(H-H_obs)^2;
end
```

UPPSALA UNIVERSITET

Task 2

- Same procedure as before - using n = find optimum 2.7633
- 1e-25 is a local optimum as see in picture
  - problem small step size
- Can't reach intervalum [1e-23,1e-27] from outside - no step here
- Use options to run algorithm longer (tol, evaluation)

Starting value	Stopping value
~1e-25	same
1	-4.4409e-16
10	1.7764e-15
100	1.4211e-14
0.1	-6.7307e-015




```
%% Task 2
options =
optimoptions(@fminunc, 'Display','iter','Algorithm','quasi-newton',
'OptimalityTolerance',1e-100, 'MaxFunctionEvaluations',
1.000000e+10);
[A_opt, f2]=fminunc(@f_A,10^(-25),options)
%Plot
x=0:10^(-26):10^(-24);
fx=arrayfun(@(x) f_A(x), x);
plot(x,fx, A_opt,f_A(A_opt), "o");
```

```
function f = f_A(A)
n = 2.7633; % set n_opt
load('Sharad.mat');
c=-2*A./(n+2);
c=c*(rho*g)^n;
s=abs(dhdx).^(n-1).*dhdx;
a_guess=(c*s);
H=(a./a_guess).^(1/(n+2));
f=norm(H-H_obs)^2;
end
```

UPPSALA UNIVERSITET

Task 3

- Some data from task 1 and task 2 with different starting values
- Demonstrate `OptimalityTolerance` and `MaxFunctionEvaluations`
- Note: When optimizing A we have a problem with the small step size, thus we can’t reach the local optimum 1e-25 do to numerical limitations - we didn’t find options to set min step size manually

Task 1 - no options		
Start	**Final point**	**Value**
0	1	1.6596e+12
1	2.6427	1.2001e+06
2	2.7633	5.3368e+05
3	2.7633	5.3368e+05
4	2.7633	5.3368e+05
5	2.7633	5.3368e+05
1.5	2.7627	5.3369e+05
1.6	2.7630	5.3368e+05

Task 1 - tolerance 1e-10		
Start	**Final point**	**Value**
0	1.5703	7.4112e+09
1	2.7633	5.3368e+05
2	2.7633	5.3368e+05
3	2.7633	5.3368e+05
4	2.7633	5.3368e+05
5	2.7633	5.3368e+05
1.5	2.7633	5.3368e+05
1.6	2.7633	5.3368e+05

Task 2 - tol 1e-6, n = 2.7633			
start	**Max eval**	**Final point**	**Value**
10	1	10	7.1386e+06
10	1e2	1.7231e-13	7.1031e+06
10	1e3	1.7764e-15	7.0460e+06
100	1e3	1.4211e-14	7.0787e+06

UPPSALA UNIVERSITET

Task 2 - with exponential function

- If we want to optimize in the range [0,1e-20], than we can use the exponential
- instead of optimizing for `f_A(x)` we can optimize for `f_A(1e-x)` this way lose the small values
- Modify our code to `fminunc(@(x) f_A(10^(-x)),start_val)` and we show the results below

Starting point	**Find optimum point**
1e15	1e-25.0037
1	1e-25.0037
0.1	1e-25.0037
1e-20	1e-25.0037
1e-24	1e-25.0037
1e-25	1e-25.0037
1e-30	1e-25.0037
1e-40	1e-26.1080
1e-50	1e-35.4569
[-1e10,-1e-10]	-1e-24.5168

Value at optimum: 5.3365e+05

Value at optimum: 3.8209e+06

Value at optimum: 1.5972e+11

Value at optimum: 3.0136e+06

UPPSALA UNIVERSITET

Task 4

Finding A and n at the same time?

```
A0 = 1e-25;
n0 = 1.1;
z0 = [n0 A0]; % both n and A in 'one' optimization variable
[z, f] = fminunc(@fmin_z,z0,opts,rho,g,dhdx,a,H_obs);

function f = fmin_z(z,rho,g,dhdx,a,H_obs)
    n = z(1);
    A = z(2);
    H = (-(n+2)*(rho*g)^(-n) * abs(dhdx).^(1-n).*a./(2*A .*dhdx)).^(1/(n+2));
    f = norm(H-H_obs)^2;
end
```

- Results:

  $n = 1.1 + \varepsilon$

  $A = 1$

- With optimizing exponent:

  $n \approx 2.6735$

  $A \approx 1e\text{-}24.75$

Task 5

- Results vary a lot → not robust
- When optimizing the exponent of A, more stable, but not good for n>>1

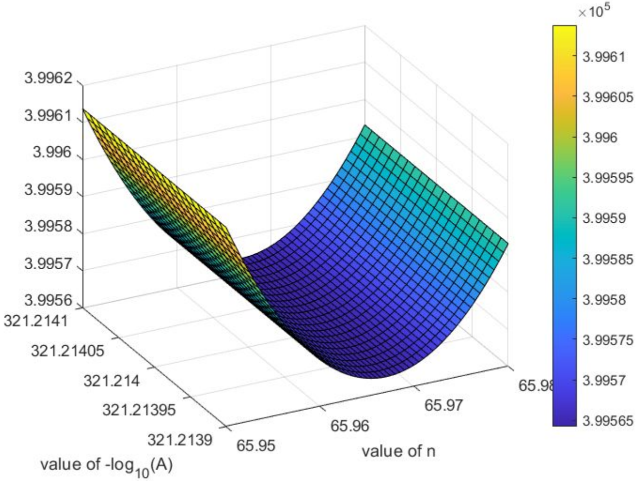
Starting point z0 = [n0, A0]	**‘Optimum’ z = [n, A]**	**Exponent ‘Optimum z = [n, A]**
[1 .1 , 1e-25]	[1.1 + ε , 1]	[2.6735 , 1e-24.75]
[1 , 1e-25]	[1.1+ ε , 1]	[2.6045 , 1e-24.75]
[1.1 , 1e-26]	[1.1+ ε , 1]	[2.7556 , 1e-25.75]
[2.7633 , 1e-25]	[2.7633 , 1e-25]	[65.533 , 1e-318.87]
[2.7633 , 1e-27]	[2.7633 , 0.111]	[65.2852 , 1e-317.95]
[1 , 1e-23]	[1 + ε , 1]	[2.2793 , 1e-22.79]

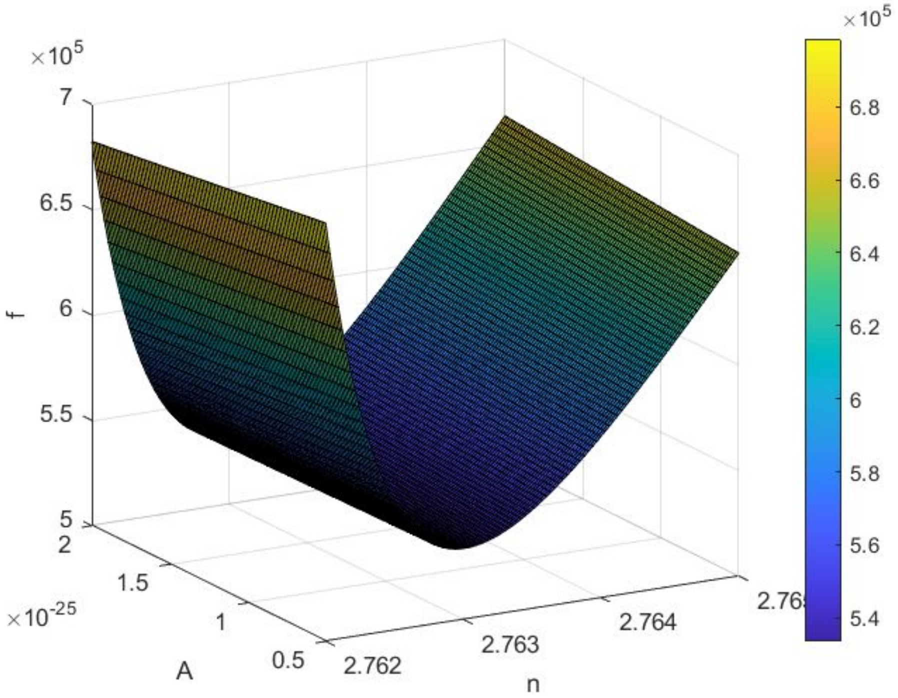
UPPSALA
UNIVERSITET

Task 6

- Objective function f(n,A)
- No local minimum?

➔ “Flat” at the bottom





UNIVERSITET

Task 7

```
%% Task 7
x = lsqnonlin(@f_n7,50)
x = lsqnonlin(@f_A7,10)
x = lsqnonlin(@(x) f_A7(10^(-x)),-15)
x = lsqnonlin(@(x)
f_An7([10^(-x(1)),x(2)]),[25,1.1])
```

- To use `lsqnonlin` we only need to modify our functions a bit - set return value f instead of ||f||^2 and use code `lsqnonlin(@modified_func,start_val)`
- So we did this for Task 1-4

Task 1 (3) - `lsqnonlin` : converge to 2.7633 from range [-1,50]

Task 2 (3)- `lsqnonlin` :

Starting value	**Stopping value**
~1e-25	same
1	3.5763e-7
10	1.7764e-015
0.1	-6.9037e-009 -113.5589e-009i

Task 2 (3)- `lsqnonlin exp`: converge to 1e-25.0037 from range [1e-122,1e14]

Task 4- `lsqnonlin exp`: starting point (A = 1e−25, n = 1.1) ==> (A=1e-321.2140, n=65.9670) stopped because “last change is smaller than function tolerance”

Function value here: **3.9956e+05**

UPPSALA UNIVERSITET