553.481/681 Numerical Analysis – Homework 2 Solutions

Problem 1. In MATLAB we rewrite `bisect.m` as `bisect2.m`, `newton.m` as `newton2.m`, and so on. We find the asymptotic error parameters by fitting over straight-line regions in the plot of $ln(e_n + 1)$ vs $\ln(e_n)$.

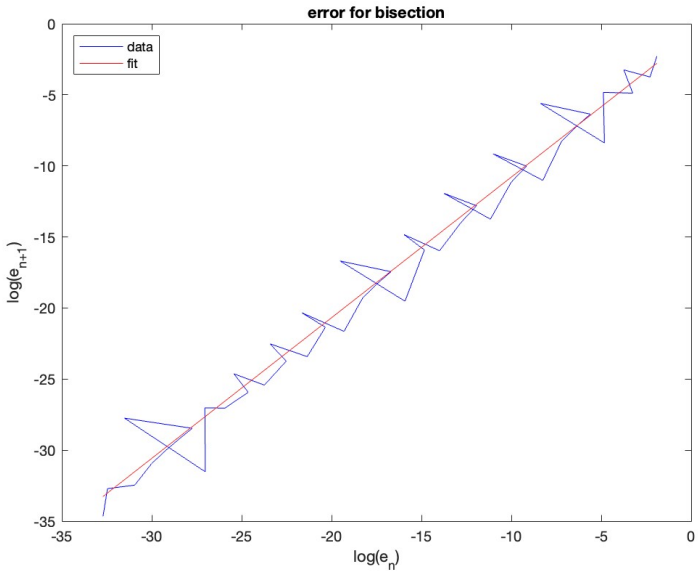
```
f=@(x) x.^3-13;
Df=@(x) 3*x.^2;
DDf=@(x) 6*x;
DDDf=@(x) 0*x+6;
xt=13^(1/3);

a=2; b=3;
tol=1e-15;
disp('bisection␣method')
bisect2
k=k

e=abs(xt-xit(1:k-3));
en=abs(xt-xit(2:k-2));
le=log(e); len=log(en);
PP=polyfit(le,len,1);
figure
plot(le,len,'-b',le,polyval(PP,le),'-r')
xlabel('log(e_n)')
ylabel('log(e_{n+1})')
legend('data','fit','location','northwest')
title('error␣for␣bisection')
p=PP(1)
pt=1
c=exp(PP(2))
ct=1/2
pause

x=2;
clear xit
disp('␣')
disp('newton␣method')
newton2
x=x
k=k

e=abs(xt-xit(1:k-2));
en=abs(xt-xit(2:k-1));
le=log(e); len=log(en);
PP=polyfit(le,len,1);
figure
plot(le,len,'-b',le,polyval(PP,le),'-r')
xlabel('log(e_n)')
ylabel('log(e_{n+1})')
legend('data','fit','location','northwest')
title('error␣for␣newton')
p=PP(1)
pt=2
c=exp(PP(2))
```

```
ct=DDf(xt)/Df(xt)/2
pause

a=2;
clear b
disp('␣')
disp('secant␣method')
secant2
x=b
k=k

e=abs(xt-xit(1:k-3));
en=abs(xt-xit(2:k-2));
le=log(e); len=log(en);
PP=polyfit(le,len,1);
figure
plot(le,len,'-b',le,polyval(PP,le),'-r')
xlabel('log(e_n)')
ylabel('log(e_{n+1})')
legend('data','fit','location','northwest')
title('error␣for␣secant')
p=PP(1)
phi=(1+sqrt(5))/2;
pt=phi
c=exp(PP(2))
ct=(DDf(xt)/Df(xt)/2)^(pt-1)
pause

a=2;
clear b
clear c
disp('␣')
disp('iquadi␣method')
iquadi2
x=c
k=k

e=abs(xt-xit(1:k-4));
en=abs(xt-xit(2:k-3));
le=log(e); len=log(en);
PP=polyfit(le,len,1);
figure
plot(le,len,'-b',le,polyval(PP,le),'-r')
xlabel('log(e_n)')
ylabel('log(e_{n+1})')
legend('data','fit','location','northwest')
title('error␣for␣iquadi')
p=PP(1)
rr=roots([-1 1 1 1]);
pt=rr(1)
c=exp(PP(2))
ct=(DDDf(xt)/Df(xt)/6)^((pt-1)/2)
```

(a) Since the error should halve every step, we expect $p = 1$ and $c = 0.5$. We implement `bisect2.m` as

```
tic

itmax=100;

if sign(f(a))==sign(f(b))
   'failure to bracket root'
   return
end

k=0
[a b b-a]
xit=[];

while abs(b-a)>tol*max(abs(b),1.0)
     if k+1>itmax
       break
     end
     x=(a+b)/2;
     xit=[xit;x];
     if sign(f(x)) == sign(f(b))
        b=x;
     else
        a=x;
     end
     k=k+1
     [a b b-a]
end
x=(a+b)/2
xit=[xit;x];

toc
```

From output of the above code, we find $c \approx 0.4095$ and $p \approx 0.9895$ by fitting the plot of $\ln(e_{n+1})$ vs. $\ln(e_n)$ with a straight line of slope p and intercept $\ln(c)$, as shown below.



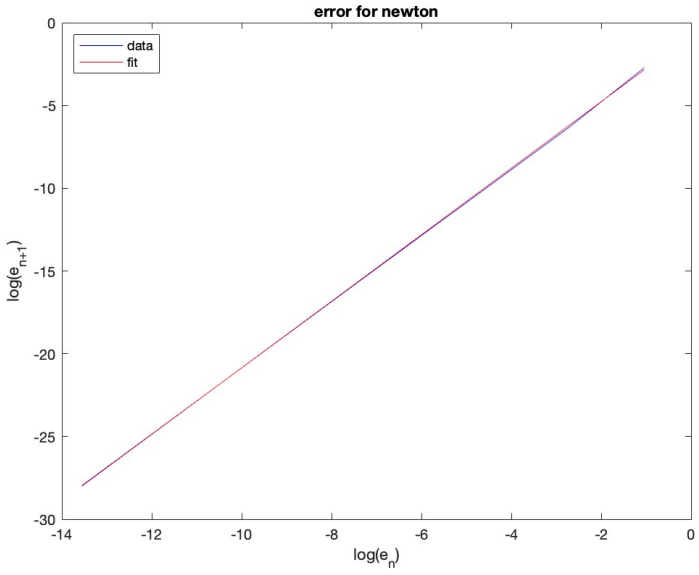
(b) Newton's method has quadratic convergence, so we expect $p = 2$. Also we expect

$$c = |\frac{f''(x^*)}{2f'(x^*)}| = \frac{(6x)_{x^*=\sqrt[3]{13}}}{2(3x^2)_{x^*=\sqrt[3]{13}}} = \frac{1}{x_{x^*=\sqrt[3]{13}}} \approx 0.4252903703$$

We implement `newton2.m` as

```
tic

itmax=100;

k=0;
if x ~= 0
  xold=0;
else
  xold=1;
end
[x abs(x-xold)];
xit=x;

while abs(x-xold)>tol*max(abs(x),1.0)
     if k+1>itmax
       break
     end
     xold=x;
     k=k+1;
     x=x-f(x)/Df(x);
     [x abs(x-xold)];
     xit=[xit;x];
end

[x abs(x-xold)];

toc
```

Numerically we find that $p \approx 2.0102$ and $c \approx 0.4729$.



(c) The secant method converges with a power of the golden ratio, so we expect

$$p = \varphi := (1 + \sqrt[3]{13})/2 \approx 1.675667344$$

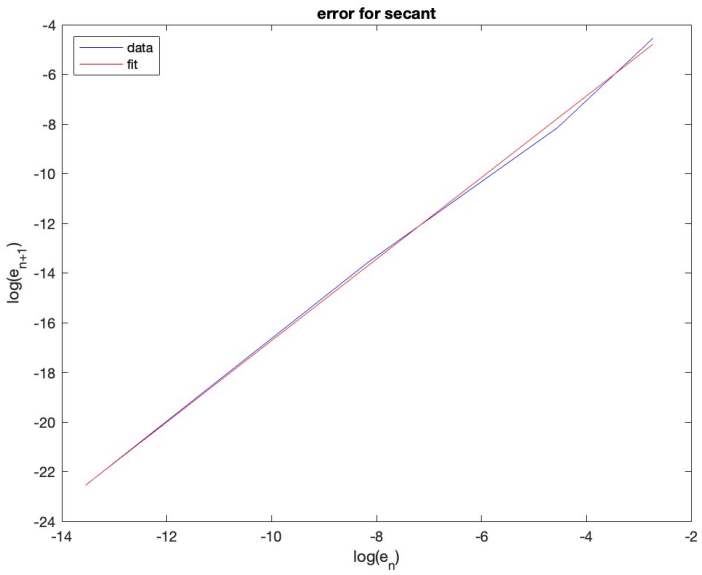
Also we expect

$$c = |\frac{f''(x^*)}{2f'(x^*)}|^{\varphi-1} = |\frac{(6x)_{x^*=\sqrt[3]{13}}}{2(3x^2)_{x^*=\sqrt[3]{13}}}|^{\varphi-1} = |\frac{1}{\sqrt[3]{13}}|^{\varphi-1} \approx 0.5611964967$$

. We implement `secant2.m` as

```
tic

k=0;

if exist('b')==0
b=a-f(a)/Df(a); k=k+1;
end
xit=b;

itmax=50;

k;
[b abs(b-a)];
fa=f(a);

while abs(b-a)>tol*max(abs(b),1.0)
     if k+1>itmax
       break
     end
     fb=f(b);
     x = b + (b-a)/(fa/fb-1);
     k=k+1;
     a=b;
     fa=fb;
     b=x;
     [b abs(b-a)];
     xit=[xit;b];
end

[b abs(b-a)];

toc
```

Numerically we find that $p \approx 1.6411$ and $c \approx 0.7319$.



(d) The inverse quadratic interpolation method converges with rate p given by the positive root of x^3-x^2-x-1, or $p \approx 1.839286755214161$, and with

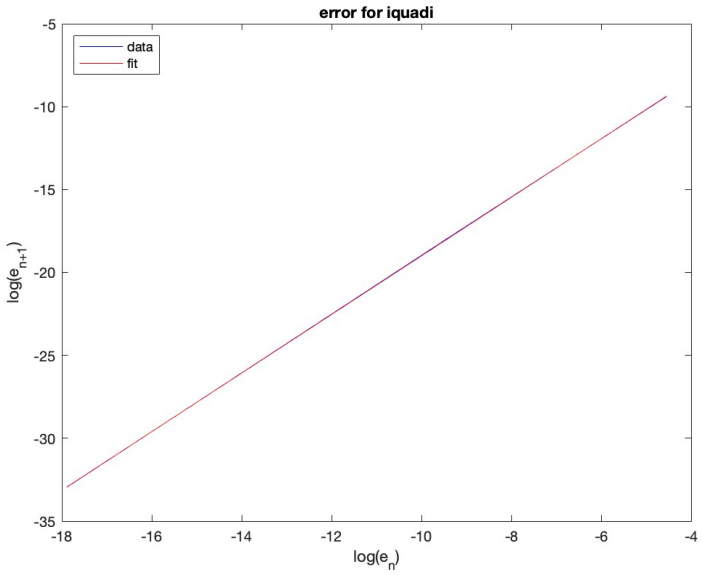
$$c = \left|\frac{f'''(x^*)}{6f'(x^*)}\right|^{(p-1)/2} = \left|\frac{6}{6(3x^2)_{x^*=\sqrt[3]{13}}}\right|^{(p-1)/2} = \left|\frac{\sqrt[3]{13}}{39}\right|^{(p-1)/2} \approx 0.307708806$$

Next `iquadi2.m` can be implemented as follows:

```
tic

itmax=100;

k=0;

fa=f(a);
if exist('b')==0
b=a-fa/Df(a); k=k+1;
end

fb=f(b);
if exist('c')==0
c = b + (b-a)/(fa/fb-1); k=k+1;
end
xit=c;

k;
[a b c];


while abs(c-b)>tol*max(abs(c),1.0)
     if k+1>itmax
       break
     end
     fc=f(c);
```

```
        u=fb/fc;
        v=fb/fa;
        w=fa/fc;
        p=w*(u-w)*(c-b)-(1-u)*(b-a);
        p=v*p;
        q=(u-1)*(v-1)*(w-1);
        x=b+p/q;
        k=k+1;
        a=b;
        fa=fb;
        b=c;
        fb=fc;
        c=x;
        xit=[xit;c];
        [c abs(c-b)];
    end

    [c abs(c-b)];

    toc
```

Numerically we find that $p \approx 1.7666$ and $c \approx 0.2634$.



Problem 2. Denote the interval in which we search by $[\frac{3\pi}{2} - \epsilon, \frac{3\pi}{2} + \epsilon]$ for some small $\epsilon > 0$. Let $M = \frac{\max_{|x-3\pi/2|\leq\epsilon} |f''(x)|}{2\cdot\max_{|x-3\pi/2|\leq\epsilon} |f'(x)|}$. We know that to guarantee convergence, $|x_* - x_0| < \frac{1}{M}$, and further, $f' \neq 0$ in this interval. We have

$$f'(x) = 2\cos(2x) + 99\cos(99x)$$
$$f''(x) = -4\sin(2x) - 99^2\sin(99x).$$

Numerically we find the nearest zero of f' around $\frac{3\pi}{2}$ to be 4.712184905708748, giving $\delta = |4.712184905708748 - 3\pi/2| \doteq 2.04074676 \times 10^{-4}$.
We also evaluate $f'\left(\frac{3\pi}{2}\right) = -2$, $f''\left(\frac{3\pi}{2}\right) = -9801$. So for ϵ small enough, $M \doteq \frac{9801}{4}$ and $\frac{1}{M} = \frac{4}{9801} \doteq 4.0812162 \times 10^{-4}$.

As a result, we require that

$$\epsilon < \min\{\delta, \frac{1}{M}\} = \delta \approx 2.04074676 \times 10^{-4}$$

The following code can be used to check when the convergence fails:

```
f=@(x) sin(2*x)+sin(99*x)-1;
Df=@(x) 2*cos(2*x)+99*cos(99*x);
DDf=@(x)-4*sin(2*x)-99^2*sin(99*x);
x=3*pi/2;
Mstar=abs(DDf(x)/Df(x)/2)
delstar=1/Mstar
pause

tol=1e-15;
M=Mstar;

figure; fplot(@(x) [Df(x), 0*x], [3*pi/2-1/M,3*pi/2+1/M])
hold on
plot(3*pi/2,0,'r*')
y=fzero(@(x) Df(x), [3*pi/2-0.01,3*pi/2+0.01]);
plot(y,0,'b*')
delta=abs(y-3*pi/2)
pause

for ii=1:200
x=3*pi/2-ii/200/M;
newton
check=abs(x-3*pi/2);
if check>tol
   fprintf('failure␣for␣%0.0f␣\n', ii)
   fprintf('failed␣at␣distance␣%0.8f␣\n', ii/200/M)
   return
end
end
```

If we run this, we see that convergence fails for the first time when $|x_0 - x_*| = 0.00020406$. This agrees quite well with our *a priori* estimate.

Problem 3. In MATLAB we first write a code called "iterate.m":

```
tic

itmax=5000;
% inputs x, tol

c=[x];

k=1
xn=g(x);
[x xn-x]
c=[c;xn];

while abs(xn-x)> tol*max(abs(x),1.0)
     if k+1>itmax
       break
```

```
      end
      x=xn;
      xn=g(x);
      k=k+1
      [x xn-x]
      c=[c;xn];
end

toc
```

Then we write the code for this problem in a separate file:

```
g=@(x) log(1+sqrt(cosh(x)));
fplot(@(x) [g(x),x], [0 1])

x=.1;
tol=5e-16
iterate
xs=x;
Dg=@(x) sinh(x)./(sqrt(cosh(x))+cosh(x))/2;
lam= Dg(xs)

xi=single(c(1:8))

x=xi(3:8);
xp=xi(2:7);
xpp=xi(1:6);

lamest=(x-xp)./(xp-xpp);
errest=lamest.*(x-xp)./(lamest-1);
xx=x-errest;

lamestf=lamest(6) % 0.1714286
lam % 0.171186758128549

errestf=errest(6) % -1.6278235e-06
error=x(6)-xs % -1.6043689e-06


impestf=xx(6) % 0.7618048
xss=single(xs)  % 0.7618048
xs=xs
imperrf=xx(6)-xs % 4.9564837e-09
```

(a) We estimate

$$\lambda_8 = \frac{(x_8 - x_7)}{(x_7 - x_6)} \approx 0.1714286$$

The true $\lambda_* = 0.171186758128549$.

(b) Apply the asymptotic error estimate: $(x_* - x_8) \approx \lambda_8(x_8 - x_7)/(1 - \lambda_8) \approx -1.6278235 \times 10^{-06}$, with the latter approximation using λ_8 from (a). The true $(x_* - x_8)$ is $-1.6043689 \times 10^{-06}$.

(c) Using the error estimate from (b) : $\hat{x}_* := x_8 + (x_* - x_8) \approx 0.7618048$. This agrees to seven digits with the exact answer $x_* = 0.761804754545927$ and greatly improving upon x_8, which is only accurate to three digits. Hence, almost single-precision accuracy is achieved here by Aitken extrapolation!

Problem 4.

(a) We first calculate the derivatives of h:

$$h'(x) = \frac{f(x)f''(x)}{(f'(x))^2}$$
$$h''(x) = \frac{f''(x)}{f'(x)} + f(x)\frac{d}{dx}\left[\frac{f''(x)}{(f'(x))^2}\right]$$

so $h'(x_*) = 0$ and $h''(x_*) = f''(x_*)/f'(x_*)$. Now we calculate the derivatives of g,

$$g'(x) = h'(x)\left(1 - \frac{f'(h(x))}{f'(x)}\right) + \frac{f(h(x))f''(x)}{f'(x)^2}$$
$$g''(x) = h''(x)\left(1 - \frac{f'(h(x))}{f'(x)}\right) + h'(x)\frac{d}{dx}\left(1 - \frac{f'(h(x))}{f'(x)}\right)$$
$$+ f'(h(x))h'(x)\frac{f''(x)}{f'(x)^2} + f(h(x))\frac{d}{dx}\left(\frac{f''(x)}{f'(x)^2}\right)$$

So that $g'(x_*) = 0$ and

$$g''(x_*) = h''(x) \times 0 + 0 \times \frac{d}{dx}\left(1 - \frac{f'(h(x))}{f'(x)}\right)|_{x=x_*} + 0 \times f'(x)\frac{f''(x)}{f'(x)^2} + 0 \times \frac{d}{dx}\left(\frac{f''(x)}{f'(x)^2}\right) = 0!$$

Since $g(x_*) = x_* \neq 0$ but $g'(x_*) = g''(x_*) = 0$, we can apply the theorem proved in class (pg 6 of Ch. 2 of the notes), to say that we have order of convergence at least 3.

(b) The convergence times for Newton and the 3rd order method we have here are:

$$\tau_{newt} = (\tau_f + \tau_{f'})\frac{\log K}{\log 2}$$
$$\tau_{3rd} = (2\tau_f + \tau_{f'})\frac{\log K}{\log 3}$$

Thus

$$\frac{\tau_{3rd}}{\tau_{newt}} = \left(\frac{2\tau_f + \tau_{f'}}{\tau_f + \tau_{f'}}\right)\frac{\log 2}{\log 3} < 1 \quad \text{iff} \quad \left(\frac{2\tau_f + \tau_{f'}}{\tau_f + \tau_{f'}}\right) < \frac{\log 3}{\log 2} = \log_2 3$$
$$\text{iff} \quad \frac{\tau_{f'}}{\tau_f} > \frac{2 - \log_2 3}{\log_2 3 - 1} \approx 0.70951129235$$

Problem 5. (a) Write

$$\left(A + xy^\top\right)\left(A^{-1} - \frac{A^{-1}xy^\top A^{-1}}{1 + \langle y, A^{-1}x\rangle}\right) = I + xy^\top A^{-1} - \frac{xy^\top A^{-1} + xy^\top A^{-1}xy^\top A^{-1}}{1 + \langle y, A^{-1}x\rangle}$$
$$= I + xy^\top A^{-1} - \frac{xy^\top A^{-1} + x\langle y, A^{-1}x\rangle y^\top A^{-1}}{1 + \langle y, A^{-1}x\rangle}$$
$$= I + xy^\top A^{-1} - \frac{xy^\top A^{-1}(1 + \langle y, A^{-1}x\rangle)}{1 + \langle y, A^{-1}x\rangle}$$
$$= I + xy^\top A^{-1} - xy^\top A^{-1} = I$$

(b) Recall that $A_n = A_{n-1} + (\Delta f_{n-1} - A_{n-1}\Delta x_{n-1})\Delta x_{n-1}^\top / \|\Delta x_{n-1}\|_2^2$. Use the Sherman–Morrison formula to write

$$\begin{aligned}
A_n^{-1} &= A_{n-1}^{-1} - \frac{A_{n-1}^{-1}(\Delta f_{n-1} - A_{n-1}\Delta x_{n-1})\Delta x_{n-1}^\top A_{n-1}^{-1}/\|\Delta x_{n-1}\|_2^2}{1 + \left\langle \Delta x_{n-1}, A_{n-1}^{-1}(\Delta f_{n-1} - A_{n-1}\Delta x_{n-1})/\|\Delta x_{n-1}\|_2^2 \right\rangle} \\
&= A_{n-1}^{-1} - \frac{A_{n-1}^{-1}(\Delta f_{n-1} - A_{n-1}\Delta x_{n-1})\Delta x_{n-1}^\top A_{n-1}^{-1}}{\|\Delta x_{n-1}\|_2^2 + \left\langle \Delta x_{n-1}, A_{n-1}^{-1}\Delta f_{n-1} - \Delta x_{n-1} \right\rangle} \\
&= A_{n-1}^{-1} - \frac{A_{n-1}^{-1}(\Delta f_{n-1} - A_{n-1}\Delta x_{n-1})\Delta x_{n-1}^\top A_{n-1}^{-1}}{\left\langle \Delta x_{n-1}, A_{n-1}^{-1}\Delta f_{n-1} \right\rangle} \\
&= A_{n-1}^{-1} + \frac{(\Delta x_{n-1} - p_{n-1})\Delta x_{n-1}^\top A_{n-1}^{-1}}{\langle \Delta x_{n-1}, \Delta p_{n-1} \rangle} \quad \text{with } \Delta p_{n-1} := A_{n-1}^{-1}\Delta f_{n-1}
\end{aligned}$$

.

Problem 6. Note that

$$\mathbf{Df}(\mathbf{x}) = \begin{bmatrix} 2x & 3y^2 & -e^z \\ 2(x-2) & z & y \\ yz & xz & xy + 2z \end{bmatrix}.$$

In MATLAB:

```
f=@(x) [x(1)^2+x(2)^3-exp(x(3)); ...
            (x(1)-2)^2+x(2)*x(3)-2; ....
            x(1)*x(2)*x(3)+x(3)^2-2];
Df=@(x)[2*x(1) 3*x(2)^2 -exp(x(3)); ...
              2*(x(1)-2) x(3) x(2); ....
              x(2)*x(3) x(1)*x(3) x(1)*x(2)+2*x(3)];

tol=eps
timnewt=0;
timbroy=0;

for ntry=1:1e5

x=[1;1;1];
tic
[xnewt,itnewt]=newtraph(f,Df,x,tol);
timnewt=timnewt+toc;

x=[1;1;1];
tic
[xbroy,itbroy]=broyden(f,Df,x,tol);
timbroy=timbroy+toc;

end

xnewt=xnewt
itnewt=itnewt
timnewt=timnewt/ntry
```

```
xbroy=xbroy
itbroy=itbroy
timbroy=timbroy/ntry

timrat=timnewt/timbroy

dx=xnewt-xbroy
```

With $N_{\text{repeat}} = 10^5$, Newton–Raphson runs in 5 iterations, Broyden's in 10. Relative mean clock-time of Newton–Raphson to Broyden's is 3.2631.

Problem 8. (a) The steepest descent direction is

$$\mathbf{d}_n = -\nabla\Phi(\mathbf{x}_n)$$

Now taking the partial derivative of $\phi(x)$ in terms of x_i, we see that

$$\frac{\partial}{\partial x_i}\Phi(\mathbf{x}) = \frac{1}{2}\sum_{j=1}^{d}\frac{\partial}{\partial x_i}f_j(\mathbf{x})^2 = \sum_{j=1}^{d}\mathbf{Df}(\mathbf{x})_{ji}f_j(\mathbf{x}),$$

So therefore we get that

$$\mathbf{d}_n = -\mathbf{Df}(\mathbf{x}_n)^\top\mathbf{f}(\mathbf{x}_n)$$

(b) In order to show this we evaluate the following inner product:

$$\begin{aligned}\langle\Delta\mathbf{x}, \nabla\Phi(\mathbf{x})\rangle &= -\langle\mathbf{Df}(\mathbf{x})^{-1}\mathbf{f}(\mathbf{x}), \mathbf{Df}(\mathbf{x})^\top\mathbf{f}(\mathbf{x})\rangle \\ &= -\langle\mathbf{f}(\mathbf{x}), (\mathbf{Df}(\mathbf{x})^{-1})^\top\mathbf{Df}(\mathbf{x})^\top\mathbf{f}(\mathbf{x})\rangle \\ &= -\|\mathbf{f}(\mathbf{x})\|_2^2 \\ &\leq 0\end{aligned}$$