Problem 1

Let $f(x, y)$ be a smooth function such that $f(x, y) = 0$ defines a smooth curve in the $x, y$-plane. We want to find some point on this curve that lies in the neighborhood of a start guess $(x_0, y_0)$ that is off the curve. One might consider fixing $y = y_0$ and solve for $x$, such that $f(x, y_0) = 0$ (or similarly fix $x = x_0$ and solve for $y$). This would however not be good strategies, since the iterations may lead to points on the curve far away from the starting point (or unnecessarily fail to converge).

a. Derive the iteration scheme

$$\begin{align*}
x_{n+1} &= x_n - df_x \\
y_{n+1} &= y_n - df_y
\end{align*}$$

for solving the task outlined above. Here, $d = \frac{f}{f_x^2 + f_y^2}$, and it is understood that $f, f_x, f_y$ are all evaluated at $(x_n, y_n)$.

Hint: One way to proceed is to look for a new iterate $(x_{n+1}, y_{n+1})$ that (i) lies on the gradient line through $(x_n, y_n)$ and (ii) also obeys $f(x, y) = 0$. Apply Newton to this $2 \times 2$ system.

b. The iteration scheme above generalizes in an obvious way to moving a point from a start location $(x_0, y_0, z_0)$ onto a surface $f(x, y, z) = 0$. With use of this iteration, find a point on the ellipsoid $x^2 + 4y^2 + 4z^2 = 16$ when starting from $x_0 = y_0 = z_0 = 1$. Give numerical evidence showing that the iteration indeed is quadratically convergent.

Solution

a. If I were to define the vector form of $f(x, y) := f(r)$, then Newton’s method would become

$$r_{n+1} = r_{n+1} - \frac{f}{||\nabla f||^2} \nabla f$$

If I convert back to Cartesian then

$$\begin{align*}
x_{n+1} &= x_n - \frac{f_x}{f_x^2 + f_y^2}f_x \\
y_{n+1} &= y_n - \frac{f_x}{f_x^2 + f_y^2}f_y
\end{align*}$$

b. Implemented in Mathematica. I found that the digits of precision approximately double suggesting quadratic convergence.
\[
\]
\[
\text{NablaF} = ((D[f[[x, y, z]]], x), D[f[[x, y, z]]], y),
\]
\[
D[f[[x, y, z]]], z) / (x \rightarrow \# [1], y \rightarrow \# [2], z \rightarrow \# [3]) \&;
\]
\[
X = N[\# - \text{NablaF}[\#]] / \text{Norm}[\text{NablaF}[\#]]^2 \&;
\]
\[
X0 = (1, 1, 1);
\]
\[
\text{Print}[(0, X0, "\"")];
\]
\[
X1 = X[X0];
\]
\[
\text{Print}[[1, X1, \text{Log10[\text{Norm[X0-X1]]}}]];\]
\[
\text{For}[\text{i = 2, i \leq 100 \& \& \text{Norm[X0-X1]} > 0, i++,
\]
\[
X0 = X1;
\]
\[
X1 = X[X0];
\]
\[
\text{Print}[[\text{i, X1, \text{Log10[\text{Norm[X0-X1]]}}]];]
\]
\[
\{0, \{1, 1, 1\}, \}
\]
\[
\{1, (1.10606, 1.42424, 1.42424), -0.215189\}
\]
\[
\{2, (1.09393, 1.36174, 1.36174), -1.04955\}
\]
\[
\{3, (1.09364, 1.36033, 1.36033), -2.69513\}
\]
\[
\{4, (1.09364, 1.36033, 1.36033), -5.98629\}
\]
\[
\{5, (1.09364, 1.36033, 1.36033), -12.5688\}
\]
\[
\{6, (1.09364, 1.36033, 1.36033), \text{Indeterminate} \}
\]

**Problem 2**

The Chebyshev expansion of arctan \(x\) over \([-1,1]\) was given in Problem 4b of Homework 8. Often, Chebyshev coefficients are not known analytically and have instead to be computed numerically. The following very brief Matlab code uses the Fast Cosine Transform method to approximate the leading coefficients in the Chebyshev expansion of any function (here applied to arctan \(x\) and set to output the first 12 coefficients):

\[
N = \ldots; \ % \text{Give here a suitable value for } N
\]
\[
x = \cos(pi *[0:2*N-1]/N);
\]
\[
a = \text{atan}(x); \ % \text{Enter here any function we want to get}
\]
\[
\% \text{the Chebyshev coefficients for}
\]
\[
f = \text{fft}(a)/N; f(1)=f(1)/2;
\]
\[
\text{real(f(1:12))) \ % \text{Print out first 12 coefficients}
\]

Run this code with \(N = 16\) and \(N = 32\) (specifying format long to get 14 decimal digits of accuracy displayed) and find out how many digits are correct in the resulting approximations for the different coefficients.

**Solution**

Knowing that the even coefficients vanish, I implemented the given method to compare the approximate coefficients to the true ones (from HW 8).

```matlab
function [err] = DCTChebErr(M)
    x = cos(pi *[0:2*M-1]/M);
    a = atan(x); \ % \text{Enter here any function we want to get}
    \% \text{the Chebyshev coefficients for}
    f = fft(a)/M; f(1)=f(1)/2;
```
```matlab
c = zeros(1,2*M);
for k = 0:M-1
    % the true coeffs
    c(2*k+1+1) = 2*(-1)'(k/(2*k+1))*(sqrt(2)-1)^(2*k+1);
end
err = zeros(1,M);
for i = 1:M
    % the even coeffs = 0
    % (with index starting at 0)
    err(i) = log10(abs(c(2*i)-f(2*i)));
end

I plotted the digits of precision for the first, second, third, ... odd coefficients, for n = 16, 23.

hold on
prec = -DCTChebErr(16);
plot(prec,'r--')
prec = -DCTChebErr(32);
plot(prec,'b--')
xlabel('odd coef')
ylabel('precision')
legend('N = 16','N = 32')
hold off
```

**Problem 3**

In the ‘Orthogonal polynomials’ notes on the class web page, the Hermite polynomials $H_n(x)$ are defined near the bottom of page 2, and given in explicit form through Rodrigues’ Formula (near the bottom of page 4). Defining $H_n(x)$ by means of Rodrigues’ Formula:

a. Determine explicitly $H_0, H_1,$ and $H_2$.

b. Verify that $H_n(x)$ satisfy the required orthogonality (including $\int_{-\infty}^{\infty} H_n(x)^2 e^{-x^2} dx = \sqrt{\pi} 2^n n!$).

c. Verify the differential equation for $H_n(x)$ (as also given on page 4 of the notes),

d. Verify the three-term recursion relation (also stated on page 4).
Solution

\[ H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} e^{-x^2} \]

a. 

\[ H_0(x) = e^{x^2} e^{-x^2} = 1 \]
\[ H_1(x) = -e^{x^2} (-2x)e^{-x^2} = 2x \]
\[ H_2(x) = e^{x^2} (-2 + 4x^2)e^{-x^2} = 4x^2 - 2 \]

b. The inner product of \( H_m \) and \( H_n \) is

\[ (H_m, H_n) = \int_{\infty}^{\infty} w H_m H_n dx \]
\[ = \int_{\infty}^{\infty} w H_m (-1)^m w^{-1} w^{(n)} dx \]
\[ = (-1)^n \int_{-\infty}^{\infty} H_m w^{(n)} dx \]

Using the second result from (c), integrating by parts, and arbitrarily assuming that \( m \leq n \) yields

\[ (H_m, H_n) = (-1)^n \int_{-\infty}^{\infty} H_m w^{(n)} dx \]
\[ = -(-1)^n \int_{-\infty}^{\infty} H_m w^{(n-1)} dx \]
\[ = (-1)^{n-1} 2m \int_{-\infty}^{\infty} H_{m-1} w^{(n-1)} dx \]
\[ = 2m (H_{m-1}, H_{n-1}) \]
\[ = \begin{cases} 
0 & m < n \\
2^n n! \int_{-\infty}^{\infty} w dx & m = n 
\end{cases} \]

The \( m < n \) case results from the fact that \( H_0 = 1 \) For the \( m = n \) case \( \int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi} \). Thus \( (H_n, H_n) = 2^n n! \sqrt{\pi} \).

c. Let \( w(x) = e^{-x^2} \). Then \( H_n = (-1)^n w^{-1} w^{(n)} \) and \( w^{(n)} = (-1)^n w H_n \).

\[ w^{(n+1)} = \frac{d^n}{dx^n} w = (-1)^n \frac{d}{dx} [w H_n] \]
\[ = (-1)^{n+1} w [2x H_n - H_n'] \]

\[ H_{n+1} = (-1)^{n+1} w^{-1} w^{(n+1)} 2x H_n - H_n' \]

Combining this with the result from (d) yields

\[ \begin{cases} 
H_n'' = 2x H_n' - 2n H_n \\
H_n' = 2n H_{n-1} 
\end{cases} \]

d. By the Leibniz rule

\[ w^{(n+1)} = \frac{d^n}{dx^n} w' = \frac{d^n}{dx^n} [-2xw] \]
\[ = -2x w^{(n)} - 2n w^{(n-1)} \]
\[ H_{n+1} = (-1)^{n+1} w^{-1} w^{(n+1)} = (-1)^n w^{-1} 2x w^{(n)} - (-1)^{n-1} w^{-1} 2n w^{(n-1)} \]
\[ = 2x H_n - 2n H_{n-1} \]
Problem 4

Starting instead with the generating function for Hermite polynomials

\[ e^{2xt-t^2} = \sum_{n=0}^{\infty} H_n(x) \frac{t^n}{n!} , \]

deduce

a. Rodrigues' formula.

b,c,d,e: Without making any further use of Rodrigues' formula (i.e. so as not to reduce the present problem to the what you just did in parts a-d), return to the generating function and directly from that re-derive the results in parts a-d.

Hints: The following are some additional results that you might find helpful to derive 'along the way':

1. \( e^{-(x-t)^2} = \sum_{n=0}^{\infty} H_n(x) e^{-x^2} \frac{t^n}{n!} \)
2. \( H'_n(x) = 2nH_{n-1}(x) \)
3. \( \int_{-\infty}^{\infty} H_{m+1}(x)H_{n+1}(x)e^{-x^2}dx = 2(m+1)\int_{-\infty}^{\infty} H_m(x)H_n(x)e^{-x^2}dx \)
   As an example of the derivation procedure when working with generating functions, consider the case of \( H'_n(x) = 2nH_{n-1}(x) \). Define \( h(x,t) = e^{2xt-t^2} = \sum_{n=0}^{\infty} H_n(x) \frac{t^n}{n!} \). Then \( \frac{\partial h}{\partial x} = \sum_{n=0}^{\infty} H'_n(x) \frac{t^n}{n!} \). Also \( h = \sum_{n=0}^{\infty} H_n(x) \frac{t^n}{n!} = \sum_{n=1}^{\infty} H_{n-1}(x) \frac{t^{n-1}}{(n-1)!} \), so \( th = \sum_{n=1}^{\infty} nH_{n-1}(x) \frac{t^n}{n!} \). Combining the results above:
   \( \frac{\partial h}{\partial x} - 2th = \sum_{n=0}^{\infty} \{H_n(x) - 2nH_{n-1}(x)\} \frac{t^n}{n!} \)

Given that \( h(x,t) = e^{2xt-t^2} \) we can immediately verify that the LHS is identically zero. Therefore, every Taylor coefficient in the RHS must vanish, i.e. \( H'_m(x) = 2nH_{n-1}(x) \) hold for \( n = 0,1,2,\ldots \)

Solution

a. Using the Taylor expansion of \( e^{2xt-t^2} \) about \( t = 0 \),

\[ e^{2xt-t^2} = \sum_{n=0}^{\infty} \frac{t^n}{n!} \frac{d^n}{dt^n} e^{2xt-t^2} \bigg|_{t=0} \]

\[ H_n(x) = \frac{d^n}{dt^n} e^{2xt-t^2} \bigg|_{t=0} = e^{x^2} \frac{d^n}{dt^n} e^{(x-t)^2} \bigg|_{t=0} \]

\[ \frac{df(x-t)}{dt} = -\frac{df(x-t)}{dx} \]

\[ H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} e^{(x-t)^2} \bigg|_{t=0} \]

\[ = (-1)^n e^{x^2} \frac{d^n}{dx^n} e^{x^2} \]
b. 
\[ H_n(x) = \left. \frac{d^n}{dt^n} e^{2xt-t^2} \right|_{t=0} \]
\[ H_0 = e^0 = 1 \]
\[ H_1 = (2x - 2t)e^{2xt-t^2} \bigg|_{t=0} = 2x \]
\[ H_1 = [(2x - 2t)^2 - 2] e^{2xt-t^2} \bigg|_{t=0} = 4x^2 - 2 \]

c. Let \( f = e^{(x-t)^2} = he^{-x^2} \). Then \( f_t = (2x - 2t)e^{(x-t)^2} = -f_x \). And
\[
(H_m, H_n) = \int_{-\infty}^{\infty} H_m \left[ \frac{d^n f}{dt^n} \right]_{t=0}^x dx
\]
\[ = \int_{-\infty}^{\infty} H'_m \left[ \frac{d^{n-1} f}{dt^{n-1}} \right]_{t=0}^x dx \]
\[ = 2m \int_{-\infty}^{\infty} H_{m-1} \left[ \frac{d^{n-1} f}{dt^{n-1}} \right]_{t=0}^x dx \]
\[ = 2m(H_{m-1}, H_{n-1}) \]

Since I have shown that \( (H_m, H_n) = 2m(H_{m-1}, H_{n-1}) \) by way of the generating function, from the latter part of the proof in 3b I know that
\[
(H_m, H_n) = \begin{cases} 0 & m \neq n \\ 2^n n! \sqrt{\pi} & m = n \end{cases}
\]

d. Substituting \( H'_n(x) = 2nH_{n-1}(x) \) into the result from (e), yields
\[
0 = H_n - 2xH_{n-1} + 2(n-1)H_{n-2}
\]
\[ = H_n - \frac{2x}{2n} H'_n + \frac{2(n-1)}{2(n-1)2n} H''_n \]
\[ = H''_n - 2xH'_n + 2nH_n \]

e. Let \( h(x, t) = e^{2xt-t^2} \). Then \( \frac{\partial h}{\partial t} = (2x - 2t)h \).
\[
\frac{\partial h}{\partial t} = \sum_{n=0}^{\infty} \frac{t^{n-1}}{n!} H_n
\]
\[ = \sum_{n=0}^{\infty} \frac{t^n}{(n-1)!} H_n
\]
\[ = \sum_{n=0}^{\infty} \frac{t^n}{n!} H_{n+1} \]
\[ = \sum_{n=1}^{\infty} 2nH_{n-1}(x) \frac{t^n}{n!} \]
\[ = \sum_{n=0}^{\infty} \left\{ H_{n+1} - 2xH_n + 2nH_{n-1} \right\} \frac{t^n}{n!} \]
\[ = H_{n+1} - 2xH_n + 2nH_{n-1} \]