Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Models, Algorithms and Data (MAD): Introduction to Computing

Spring semester 2019

Prof. Dr. Jens Honore Walther
Dr. Julija Zavadlav
ETH Zentrum, CLT
CH-8092 Zürich

Exam

Issued: Saturday, 10.08.2019

Exam directives. In order to pass the exam, the following requirements have to be met:

- Read carefully the first two pages of the exam. Write your name and Legi-ID where requested. Before handing in the exam, **PUT YOUR SIGNATURE ON PAGE 2**.
- Clear your desk (no cell phones, cameras, etc.): on your desk you should have your Legi, your pen, paper and your notes.
- If necessary the teaching assistants will give you additional paper sheets. On the top-right corner of every page write your complete name and Legi-ID.
- The personal summary consists of no more than 4 pages (2 sheets). The personal summary can be handwritten or machine-typed. In case it is machine-typed, the text has to be single-spaced and the font size has to be at least 8 pts.
- You can answer in English or in German; the answers should be handwritten and clearly readable, written in blue or black - do NOT write anything in red or green. Only one answer per question is accepted. Invalid answers should be clearly crossed out.
- If something is disturbing you during the exam, or it is preventing you from peacefully solving the exam, please report it immediately to an assistant. Later notifications will not be accepted.
- You must hand in: the exam cover, the sheets with the exam questions and your solutions. The exam cannot be accepted if the cover sheet or the question sheets are not handed back.

Family Name:	
Name:	
Legi-ID:	

Question	Maximum score	Score	TA 1	TA 2
1	3			
2	5			
3	3			
4	3			
5	3			
6	5			
7	2			
8	3			
9	12			
10	5			
11	12			
12	6			
13	12			
14	2			
15	8			
Total	84			

With your signature you confirm that you have read the exam directives; you solved the exam without any unauthorized help and you wrote your answers following the outlined directives.

Signature: ______________________

GOOD LUCK!

Short Questions [24 Points]

For each of the following exercises there is **one** correct answer. In each question the correct answer gives $+1$ point, for the wrong -1 point. The minimal number of points for your answers to all short questions is 0.

Question 1: Multi-dimensional nonlinear solver

a) Bisection, Newton and Secant methods all work well for one dimensional nonlinear problems. Which can be extended directly to multi-dimensional nonlinear problems?

- ☐ Bisection and Newton
- ☐ Bisection and Secant
- ☐ Newton and Secant
- ☐ Newton

b) What is the right ordering in terms of order of convergence (from fastest to slowest)

- ☐ Newton, Bisection, and Secant
- ☐ Bisection, Secant, and Newton
- ☐ Newton, Secant, and Bisection
- ☐ Secant, Newton, and Bisection

c) Which of the following statements is true?

- ☐ If the initial guess is around the true solution, the Newton method always has quadratic convergence rate.
- ☐ The sufficient conditions for a minimum of $E(\vec{x})$ are: $\nabla E = 0$ and the Hessian matrix of E is positive definite at x^*.
- ☐ Newton method is not applicable for a least squares problem.
- ☐ The problem of minimizing $E(\vec{x})$ is exactly the same as solving a non-linear equation $\nabla E(\vec{x}) = 0$.

Question 2: Lagrange interpolation

a) The Lagrange basis functions $l_j(x)$ are equal to zero for all $x_{k \neq j}$ and equal to one for x_j. Therefore we can also write the interpolating function as $f(x) = \sum_{k=1}^{N} y_k \delta_{kj}$, where δ_{kj} is the Kronecker delta function:

- ☐ True
- ☐ False

b) Assume you have samples from a quadratic function. How many points are required such that Lagrange interpolation exactly resembles the quadratic function that we sampled from?

- ☐ 2
- ☐ 3
- ☐ 4
- ☐ 6

c) Assume you have a dataset consisting of 5 points. How many roots does each single term of the interpolating function have, ie. $l_i(x) = 0$?

☐ 3
☐ 4
☐ 5
☐ Depends on the datapoints

d) You used Lagrange interpolation to interpolate a set of data points. However, you lose the original data set and only have an incomplete graph, figure 1. What might the data points have been?

☐ $\{(5, 1), (15, 1)\}$
☐ $\{(5, 2), (20, 10)\}$
☐ $\{(15, 9), (20, 10)\}$
☐ $\{(3.1, 1.1), (5, 2), (20, 10)\}$
☐ $\{(5, 2), (15, 9), (20, 10)\}$
☐ Cannot be said from the given information

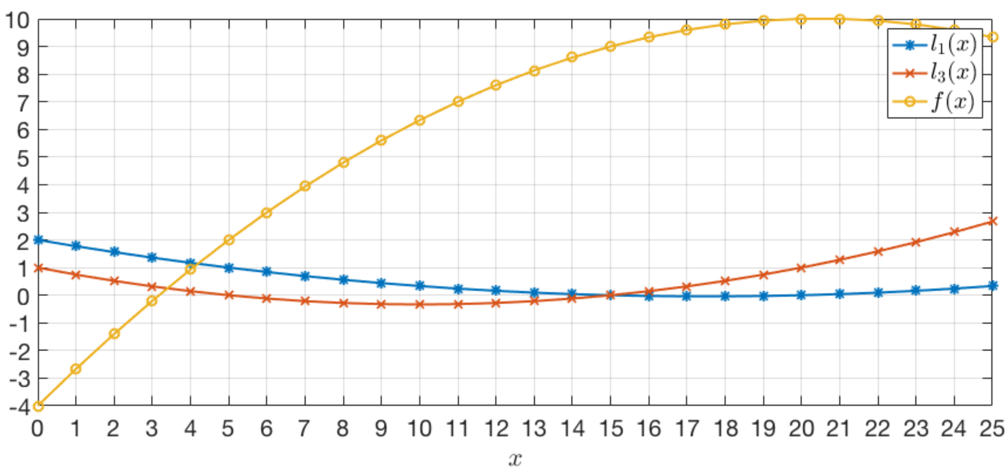


Figure 1: Lagrange interpolation $f(x)$ and two basis function $l_1(x)$ and $l_3(x)$.

e) You have an arbitrary dataset $(x_1, y_1), ..., (x_N, y_N)$ and decide to use Lagrangian interpolation:

$$f(x) = \sum_{k=1}^{N} w_k l_k(x)$$

Here $l_k(x)$ denote the Lagrange functions introduced in the lecture. We want to perform an optimization for the parameters w_k using least squares:

$$w_1, ..., w_N = \arg\min_{w_1,...,w_N} \sum_{k=1}^{N} (w_k l_k(x_k) - y_k)^2$$

Once you have obtained the weights and plugged them into your model, your model will be

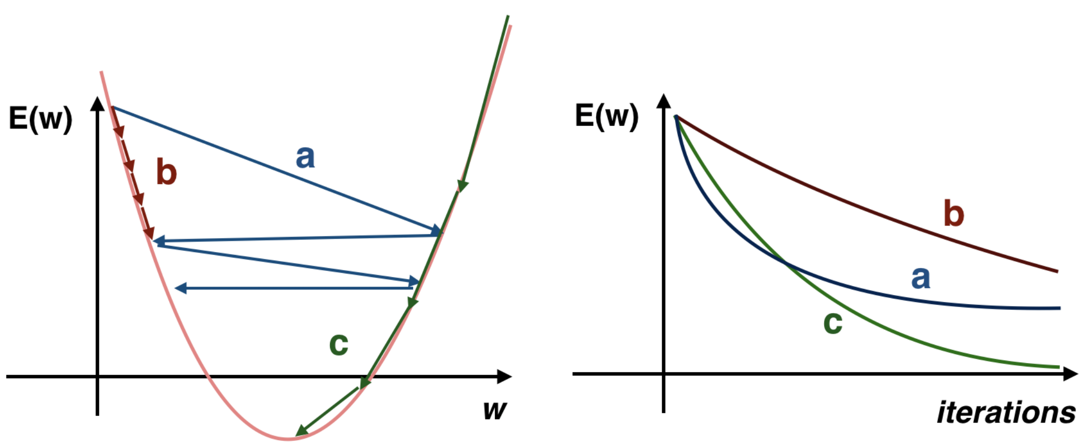
- ☐ ... the same as when you would have used Lagrange interpolation
- ☐ ... slightly different than the Lagrange interpolation due to the square term in the cost function
- ☐ ... completely different since the least square approach is not comparable with Lagrange interpolation

Question 3: Artificial Neural Network Training

Artificial neural networks are trained by stochastic gradient descent. In its most basic form, this method has a single parameter, the learning rate η.

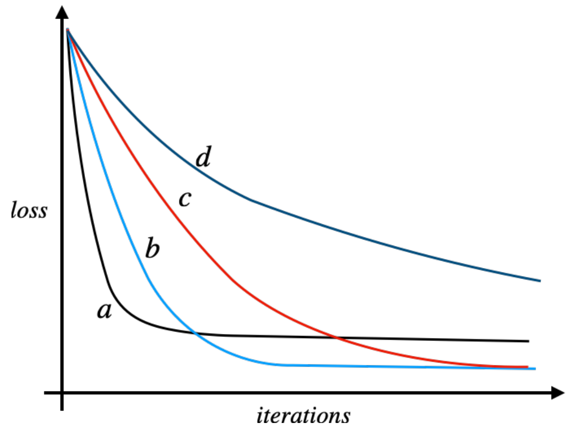
a) Each training curve on the left figure corresponds to a curve on the right figure. Each pair of curves correspond to a different learning rate. Select the correct combination of curves:

- ☐ (a,1), (b,3), (c,2)
- ☐ (a,1), (b,2), (c,3)
- ☐ (a,2), (b,1), (c,3)
- ☐ (a,2), (b,3), (c,1)



b) Each curve on the graph correspond to the training history of a network with a different learning rate. Select the curve corresponding to the best choice of the learning rate:

- ☐ a
- ☐ b
- ☐ c
- ☐ d



c) Only one of this statement about gradient descent (GD) is true. Select the true statement:

- ☐ GD is guaranteed to converge to the global optimum.
- ☐ A larger learning rate guarantees that the model will converge to an optimal solutions in fewer iterations.
- ☐ GD can converge to a local optimum.
- ☐ Multiple initialization of a same neural network trained on the same data will converge to the same optimum.

Question 4: Richardson Extrapolation

a) The Richardson extrapolation is a simple way to increase the accuracy, which can only be used to compute integrals numerically:

- ☐ True
- ☐ False

b) Suppose that $L = L(h) + k_1 h + k_2 h^2 + \ldots$ is an approximation of the value L. If Richardson extrapolation is applied once we obtain the formula $L_1(h) = L(h/2) + a[L(h/2) - L(h)]$, where a is given by

- ☐ $a = 1$
- ☐ $a = \frac{1}{2}$
- ☐ $a = \frac{1}{3}$

c) Suppose that $L = L(h) + k_1 h^2 + k_2 h^4 + O(h^6)$ is an approximation to the value L. If Richardson's extrapolation is applied twice to $L(h)$ we obtain the formula $L_2(h) = L_1(h/2) + a[L_1(h/2) - L_1(h)]$, where a is given by

- ☐ $a = 1$
- ☐ $a = \frac{1}{3}$
- ☐ $a = \frac{1}{15}$

Question 5: Romberg Integration

a) The value of an integral $\int_a^b f(x)\mathrm{d}x$ computed using the trapezoidal rule with 1, 2, and 4 segments is given as 1, $\frac{3}{2}$ and $\frac{5}{4}$ respectively. The best estimate of the integral you can find using Romberg integration is:

- ☐ $\frac{5}{3}$
- ☐ $\frac{53}{43}$
- ☐ $\frac{51}{45}$
- ☐ $\frac{7}{6}$

b) Romberg integration combines Richardson Extrapolation with:

- ☐ Trapezoidal Rule
- ☐ Midpoint Rule
- ☐ Simpson's Rule
- ☐ any Newton-Cotes formula

c) We can achieve a 3rd order accurate integration scheme when combining Richardson integration and the Trapezoidal rule:

- ☐ True
- ☐ False

Question 6: Gaussian Quadrature

a) In the method of undetermined coefficients we approximate $\int_a^b f(x)dx$ as $c_1 f(a) + c_2 f(b)$. This approximation is exact for functions f of degree:

- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4

b) Gaussian Quadrature is used to find the optimal number of quadrature points given a fixed interval $[a, b]$.

- ☐ True
- ☐ False

c) For any $a, b \in \mathbb{R}$ with $a < b$, the 3^{rd} order Gaussian quadrature integration points for the approximation of $I = \int_a^b f(x)dx$ are given by $x_1 = -0.7745966692,\ x_2 = 0.0,\ x_3 = 0.7745966692$

- ☐ True
- ☐ False

d) In Hermite interpolation, given data points x_i, y_i and y_i', the goal is to find a polynomial $f(x)$ of degree $2n-1$ that satisfies $y_i = f(x_i)$ and $y_i' = f'(x_i)$. Therefore $f(x)$ is expressed as:

$$f(x) = \sum_{k=1}^{n} U_k(x) y_k + \sum_{k=1}^{n} V_k(x) y_k'$$

In order to satisfy the constraints on the data points and their derivatives, the polynomials $U_k(x)$ and $V_k(x)$ must have the following properties:

- ☐ $U_k(x_j) = 0$, $U_k'(x_j) = \delta_{jk}$, $V_k(x_j) = 0$, $V_k'(x_j) = \delta_{jk}$
- ☐ $U_k(x_j) = \delta_{jk}$, $U_k'(x_j) = 0$, $V_k(x_j) = 0$, $V_k'(x_j) = \delta_{jk}$
- ☐ $U_k(x_j) = 0$, $U_k'(x_j) = \delta_{jk}$, $V_k(x_j) = \delta_{jk}$, $V_k'(x_j) = 0$
- ☐ $U_k(x_j) = \delta_{jk}$, $U_k'(x_j) = 0$, $V_k(x_j) = \delta_{jk}$, $V_k'(x_j) = 0$

e) To find the optimal quadrature points for $I = \int_a^b f(x)dx \approx c_1 f(x_1) + c_2 f(x_2)$, the approximation must be exact for the integration of any polynomial of order p. Select the highest possible value for p:

- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4

Question 7: Bayesian Inference

a) We denote by M our model D as our data and Θ as the parameters of our model. The posterior distribution for the parameters is given by

$$P(\Theta|D, M)$$

Using Bayes' rule to estimate the value of the posterior, which are the quantities we need to specify?

- ☐ The prior $P(D|M)$, the likelihood $P(D|\Theta, M)$ and the evidence $P(\Theta|M)$
- ☐ The prior $P(\Theta|D)$, the likelihood $P(M|\Theta, D)$ and the evidence $P(M|D)$
- ☐ The prior $P(\Theta|M)$, the likelihood $P(D|\Theta, M)$ and the evidence $P(D|M)$

b) The Laplace approximation of an arbitrary probability distribution $P(\Theta)$ is based on a Taylor expansion of its logarithm

$$L(\Theta) = \log(P(\Theta)) \approx L(\Theta') + \left.\frac{\partial L}{\partial \Theta}\right|_{\Theta'} (\Theta - \Theta') + \frac{1}{2}\left.\frac{\partial^2 L}{\partial \Theta^2}\right|_{\Theta'} (\Theta - \Theta')^2 + \mathcal{O}\left[(\Theta - \Theta')^3\right]$$

In order to obtain the Laplace approximation we now

☐ ... exponentiate the result giving us a Gaussian approximation with variance

$$\sigma^2 = -\left(\left.\frac{\partial^2 L}{\partial \Theta^2}\right|_{\Theta^*}\right)^{-1}$$

and mean located at the optimum Θ^*.

☐ ... evaluate the Taylor expansion at the optimum $\Theta' = \Theta^*$ and exponentiate the result giving us a Gaussian approximation with variance

$$\sigma^2 = -\left(\left.\frac{\partial^2 L}{\partial \Theta^2}\right|_{\Theta^*}\right)^{-1}$$

and mean located at the optimum Θ^*

☐ ... evaluate the Taylor expansion at the optimum $\Theta' = \Theta^*$ and exponentiate the result giving us a Gaussian approximation with variance

$$\sigma^2 = \left(\left.\frac{\partial^2 L}{\partial \Theta^2}\right|_{\Theta^*}\right)^{-1}$$

and mean located at the optimum Θ^*

Numerical Problems [50 Points]

The following exercises involve calculations similar to the ones performed in the lecture and the exercises. Please use one separate sheet of paper for each of the questions.

Question 8: One dimensional nonlinear solver [3 Points]

For a one-dimensional nonlinear function $f(x)$, we may use iterative numerical schemes to find its root, that is, find x^* so that $f(x^*) = 0$. Please apply Newton method to compute $\sqrt[3]{7}$. You may stop the iteration when the step size $\mathrm{tol} = \|x^k - x^{k-1}\|$ is lower than 10^{-2}. To perform the calculation you might find the following table useful:

fraction	1/12	23/12	$(23/12)^2$	$3 \times (23/12)^2$	$(23/12)^3$	41/11021
float	0.083	1.917	3.675	11.021	7.041	0.004

Question 9: Curvature of a carrot [12 Points]

You are trying to distinguish carrots from apples. For this, you take a photo of the object and detect the contour of it. Then you compute the curvature along the contour. If the curvature is higher than a certain threshold you decide that the object is a carrot. Else you classify it as an apple (see figure 2 for an illustration).

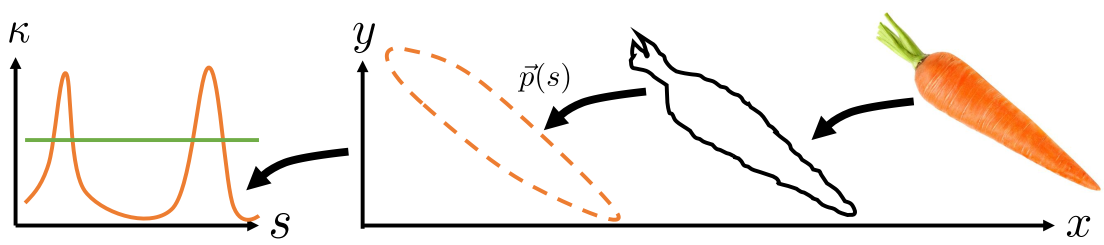


Figure 2: First the noisy contour of the image of the carrot is extracted. Then a parametrization of the contour is performed. Finally compute the curvature κ with respect to the arc-length s. The green line in the curvature plot approximately depicts the curvature of an apple.

Since the contour can be very noisy you decide to fit cubic splines. However, first you need to parametrize it by introducing s, the distance along the contour:

$$\vec{p}(s) = (x(s), y(s))$$

The starting point is irrelevant since you are only interested in the peak curvature. Note that you have to fit cubic splines to both coordinates, independently.

a) Come up with boundary conditions for the cubic splines of each coordinate considering the fact that the curvature

$$\kappa(s) = \frac{|x'(s)y''(s) - y'(s)x''(s)|}{(x'^2(s) + y'^2(s))^{3/2}}$$

has to be continuous at s_1 and s_N, since s loops around the object.

Let us assume you recover 4 points from your objects contour, $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)\}$. Please note that $(x_1, y_1) = (x_4, y_4)$, due to the loop. The corresponding s_i are are s_1, s_2, s_3, s_4. Each cubic spline writes as

$$S_i^j(s) = a_{i0}^j + a_{i1}^j s + a_{i2}^j s^2 + a_{i3}^j s^3$$

where j corresponds to the coordinate, either x or y, and $i = 1, 2, 3$ to the respective interval.

b) How many unknowns do we have to find in order to recover the full parametrization?

c) Write down all constraints for the x-coordinate. Use the general expression for the spline, i.e. $S_i^x(s)$, $S_i'^x(s)$, and $S_i''^x(s)$.

d) Now expand the functions $S_i^x(s)$, $S_i'^x(s)$, $S_i''^x(s)$ and write out all the constraints for the x-coordinate. Bring them to a format $A_x \vec{a}_x = \vec{b}_x$. Where $\vec{a}_x$ contains all the unknowns a_{i0}^x, a_{i1}^x, a_{i2}^x, a_{i3}^x for $i = 1, 2, 3$ and A_x is a matrix and $\vec{b}_x$ is a vector that you are asked to construct from the constraint you wrote down.

e) After solving the resulting linear equation you have a parametrization of the contour of your object. The next step is to compute the curvature $\kappa(s)$ in order to be able to classify the object. Assume you found $a_{10}^x = 2$, $a_{11}^x = 0$, $a_{12}^x = -1$, $a_{13}^x = 0$, $a_{10}^y = 0$, $a_{11}^y = 2$, $a_{12}^y = 0$, and $a_{13}^y = -1/3$. Compute the curvature on the arch between s_1 and s_2. To simplify your result use that $s \ll 1$ to drop high order terms.

Question 10: Orthonormal functions [5 Points]

Data is generated according to the following model:

$$y = 2x + 5x^2$$

where x is drawn from a uniform distribution on the interval $[0, 1]$, $x \sim \mathcal{U}([0, 1])$. You are going to approximate the model with a set of orthonormal functions. Since you are dealing with a probabilistic model, an appropriate inner product has to be chosen.

$$\langle h, q \rangle = \mathbb{E}_{p(x)}[hq] = \int h(x)q(x)p(x)\mathrm{d}x$$

a) Compute a set of orthonormalized function ϕ_1, ϕ_2 from $\tilde{\phi}_1 = 1$ and $\tilde{\phi}_2 = x$ using Gram-Schmidt.

b) You now want to approximate y with the two orthonormal functions. Compute α_1 and α_2 and write down the resulting $y \approx \alpha_1 \phi_1(x) + \alpha_2 \phi_2(x)$.

c) Assume that you have an additional orthonormal function ϕ_3. Show the previously computed α's do not change for the new approximation

$$y \simeq \sum_{i=1}^{3} \alpha_i \phi_i(x) \tag{1}$$

Hint: Take advantage of the fact that the inner product is linear, i.e. $\langle c \cdot h, q \rangle = c \cdot \langle h, q \rangle$, where c is a constant, and $\langle h + f, q \rangle = \langle h, q \rangle + \langle f, q \rangle$.

Question 11: Backpropagation [12 Points]

A network is built with two input neurons, one hidden layer of 3 neurons and an output layer with a single neuron. Both layers have a rectified linear unit (RELU) activation functions. This network does not have any biases.

The rectified linear unit (RELU) is defined as

$$\phi(x) = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

a) Sketch the graph of the network, make sure to label the input $\mathbf{x}$, the hidden layer $\mathbf{h}$ and the output o.

b) Write the relations between the components of the input $\mathbf{x}$, the hidden layer $\mathbf{h}$ and the output o.

c) Compute the forward pass (i.e the output o) for an input sample $\mathbf{x}^1 = (1, 2)$. The hidden weight matrix $\mathbf{W}^H$ and the output weight matrix $\mathbf{W}^O$ are initialized as follows:

$$\mathbf{W}^H = \begin{bmatrix} -3 & 1 \\ 3 & 1 \\ 2 & 2 \end{bmatrix}, \; \mathbf{W}^O = \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

d) Compute the error $E = \frac{1}{2}(\mathbf{t} - \mathbf{o})^2$ given the output o computed in the previous subquestion and the target $\mathbf{t}^1 = 14$.

e) Compute the gradient $\frac{\partial E}{\partial W_2^O}$ of the error with respect to weight W_2^O, the second weight of the output layer (use the chain rule).

f) Compute the update weight $W_{2,new}^O$ based on the old weight W_2^O using one step of gradient descent with the given learning rate $\eta = 0.1$ and the gradient $\frac{\partial E}{\partial w_2^0}$ computed in the previous subquestion.

Question 12: "Open" Newton-Cotes formula [6 Points]

The Newton-Cotes formula allows us to derive Quadrature rules by approximating the objective function by Lagrange polynomials. There exists two types of Newton-Cotes formulas, the "closed" type in which the end points of the interval are included in the quadrature (which you have learned in class), and the "open" type in which the end points are not included.

For a given order M the quadrature points for "closed" type are expressed as

$$x_i = a + i\frac{b-a}{M}, \qquad i = 0, ..., M$$

For order M the "closed" type are expressed as

$$x_i = a + i\frac{b-a}{M}, \qquad i = 1, ..., M-1$$

a) Derive the trapezoidal method of "open" type with $M = 3$ to approximate an arbitrary integral

$$I = \int_a^b f(x)\mathrm{d}x$$

b) Calculate the integral

$$J = \int_{0}^{\pi/2} \sin(x)dx$$

using the trapezoidal method of "open" type derived in the previous subquestion. *Hint:* You may find the following values useful:

x	0	$\frac{\pi}{6}$	$\frac{\pi}{4}$	$\frac{\pi}{3}$	$\frac{\pi}{2}$	π	$\frac{3\pi}{2}$	2π
$\sin(x)$	0	$1/2$	$1/\sqrt{2}$	$\sqrt{3}/2$	1	0	-1	0

c) Compare the result to the trapezoidal method of "closed "type, which is given by

$$J \approx \frac{\Delta x}{2}[f(a) + f(b)]$$

Compare the result to the previous subquestion and discuss the difference with respect to the exact solution $J = 1$.
Hint: You may find the following values useful:

x	0	$\frac{\pi}{6}$	$\frac{\pi}{4}$	$\frac{\pi}{3}$	$\frac{\pi}{2}$	π	$\frac{3\pi}{2}$	2π
$\sin(x)$	0	$1/2$	$1/\sqrt{2}$	$\sqrt{3}/2$	1	0	-1	0

Further you may use $\pi \approx 3$, $\sqrt{2} \approx 1.5$ and $\sqrt{3} \approx 1.75$.

Question 13: Monte Carlo Sampling [12 Points]

In statistical mechanics we aim to derive macroscopic thermal quantities based on probabilistic properties of the underlying microscopic systems. In the following we will regard the canonical ensemble, which is a mechanical system with fixed volume V and particle number N, which is in thermal equilibrium with a heat bath of temperature T. In this case it can be shown that the probability to find a particle with momentum $\vec{p} \in \mathbb{R}^3$ and position $\vec{q} \in \mathbb{R}^3$ corresponding to an energy $H(\vec{p}, \vec{q})$ is given by the Boltzmann (or Gibbs) distribution

$$P(\vec{p}, \vec{q}) = \frac{1}{Z} \exp\left(-\frac{H(\vec{p}, \vec{q})}{kT}\right),$$

where k is the Boltzmann constant and the normalization factor is the partition function

$$Z = C \int \exp\left(-\frac{H(\vec{p}, \vec{q})}{kT}\right) \mathrm{d}\vec{p}_1 \cdots \mathrm{d}\vec{p}_N \mathrm{d}\vec{q}_1 \cdots \mathrm{d}\vec{q}_N.$$

Here C is some constant taking into account the phase space volume and permutations of the indistinguishable particles we consider. If this integral can be solved analytically we can derive the free energy of the system as

$$F = -kT \ln(Z)$$

From this the macroscopic quantities can be derived. In most cases the integration can not be performed analytically and therefore we have to use numerical methods to approximate it.

a) Please write down the formula allowing you to integrate a function over a $6N$ dimensional space using numerical quadrature. Is this feasible if you assume that $N = 10^{23}$? Explain the associated problem using the fact that the composite Trapezoidal rule is second order accurate by deriving the error with respect to the number of sampling points.

b) Write down the formula to compute a $6N$ dimensional integral with Monte Carlo Integration. As this is a stochastic method the estimate we obtain is a random variable. Thus we have to estimate the error ϵ. A common way to do that goes via the standard deviation

$$\epsilon = \sqrt{\mathrm{Var}(\langle f \rangle_M)}$$

Show that the resulting error scales as $1/\sqrt{M}$ for Monte Carlo integration.
Hint: Use the fact that your samples are independent (i.e. $\mathbb{E}[f(x_i)f(x_j)] = \mathbb{E}[f(x_i)]\mathbb{E}[f(x_j)]$)

Monte Carlo methods depend on our ability to sample from the probability distribution governing our problem. One way to generate samples from an arbitrary distribution goes via inverse transform sampling. This method is based on the fact that samples of a random variable x with cumulative distribution function $F_X(x)$ can be obtained via samples from an uniform distribution $u \sim \mathcal{U}([0,1])$ via the following relation

$$x = F_X^{-1}(u)$$

Here F_X^{-1} denotes the inverse of the cumulative distribution function.

c) Given the exponential distribution

$$p(q) = \lambda e^{-\lambda x},$$

derive the expression allowing you to generate samples using the inverse transform method.

Pseudocode [10 Points]

In the last part of the exam you are asked to write a pseudo-code showing implementation details for some of the algorithms learned in class. As a template please us the following example computing the Fibonacci series

Algorithm 1 Fibonacci Series

Input:

N, {number of elements to compute}

$n_{\max}$, {threshold to stop computation}

Output:

$\vec{F}$, {vector containing Fibonacci numbers}

Steps:

```
F[0] ← 0
F[1] ← 1
n ← 2
while n < N + 3 do
    F[n] ← F[n − 1] + F[n − 2]
    if F[n] > n_max then
        break
    end if
    n ← n + 1
end while
```

Question 14: Linear Least Squares [2 Points]

In the year of 2050, one astronaut landed on a planet with an environment quite similar to earth in the Alpha Centauri star system, which is the closest and 4.37 light-years away from our solar system. The first task performed by the astronaut is simple: the astronaut throws a ball into the sky and records accurately a sequence of height of the ball and time elapse: (t_1, h_1), (t_2, h_2), (t_3, h_3), ... (t_N, h_N). Following Newton's law of motion, the astronaut tries to figure out the gravity/acceleration on this planet: $h = g/2t^2 + v_0 t + h_0$, where positive h is upwards, g is the gravity, v_0 and h_0 are initial velocity and position of the ball, respectively. The astronaut formulated the linear least squares for this problem and wrote a pseudocode as follows. To be more specific, the astronaut identified matrix A, unknown $\vec{x}$, and right hand side $\vec{h}$. Furthermore, the astronaut built the so-called normal equations and tried to solve them. Please help identify the mistakes/bugs made in the pseudocode.

Algorithm 2 Linear Least Squares: pseudocode

Input:

N, {number of records}
t, {vector containing time sequence}
h, {vector containing height sequence}

Output:

g, {gravity}
v_0, {initial velocity}
h_0, {initial height}

Steps:

```
i ← 1
while i ≤ N do
    A[i, 1] ← t[i] * t[i]
    A[i, 2] ← t[i]
    A[i, 3] ← 0
end while
```

Steps:

```
A_T ← transpose(A)
B ← dot product(A, A_T)
C ← dot product(A_T, h)
```

Steps:

```
D ← matrix inverse(B)
x ← dot product(C, D)
g ← x[0]
v_0 ← x[1]
h_0 ← x[2]
```

Question 15: Monte Carlo Integration [8 Points]

a) Write a pseudo code allowing you to integrate an arbitrary function using Monte Carlo Integration. Assume you are given a routine generating samples from an uniform distribution.

b) You want to estimate how well your algorithm performs. Therefore you decide to regard the acceptance rate (i.e. the number of accepted samples divided by the total number of samples). What changes do you have to do in your code to compute this measure? If you have enough space in your code from part a), please add these lines to your code. If not mark the associated places (using ⋆,# or similar) and write down the associated pseudo-codes on separate lines.

c) Assuming you found that the acceptance rate is really low. What are the two possible variants to the naive Monte Carlo Integration introduced in the lecture? Which parts of the code have to be adjusted for these improvements?