Examination Computational Statistics

Linköpings Universitet, IDA, Statistik

Course:	732A90 Computational Statistics
Date:	2023/03/22, 8–13
Teacher:	Krzysztof Bartoszek
Allowed aids:	Printed books, 100 page computer document,
Provided aids:	material in the zip file **exam_material_732A90.zip**
Grades:	A= $[18-20]$ points
	B= $[16-18)$ points
	C= $[14-16)$ points
	D= $[12-14)$ points
	E= $[10-12)$ points
	F= $[0-10)$ points
Instructions:	Provide a detailed report that includes plots, conclusions and interpretations. If you are unable to include a plot in your solution file clearly indicate the section of R code that generates it. Give motivated answers to the questions. If an answer is not motivated, the points are reduced. Provide all necessary codes in an appendix. In a number of questions you are asked to do plots. Make sure that they are informative, have correctly labelled axes, informative axes limits and are correctly described. Points may be deducted for poorly done graphs. Name your solution files as: **[your exam account id]_[own file description].[format]** If you have problems with creating a pdf you may submit your solutions in text files with unambiguous references to graphics and code that are saved in separate files. There are **TWO** assignments (with sub–questions) to solve. Provide a separate solution file for each assignment. Include all R code that was used to obtain your answers in your solution files. **PROVIDE CALLS TO YOUR CODE** Make sure it is clear which code section corresponds to which question. If you also need to provide some hand–written derivations please number each page according to the pattern: Question number . page in question number i.e. Q1.1, Q1.2, Q1.3,. . ., Q2.1, Q2.2, . . ., Q3.1, ...

NOTE: If you fail to do a part on which subsequent question(s) depend on describe (maybe using dummy data, partial code e.t.c.) how you would do them given you had done that part. You *might* be eligible for partial points.

Assignment 1 (10p)

Any positive real number, x, can be represented as

$$x = \sum_{j=0}^{\infty} b_j 2^{p-j},$$

for some p (that will depend on x) and $b_j \in \{0, 1\}$. The values of p and $\{b_j\}_{j=0}^{\infty}$ will be the binary representation of x. Of course on a computer one cannot have an infinite number of bits, but will need to approximate x as

$$x \approx \tilde{x} = 2^p + \sum_{j=1}^{m} b_j 2^{p-j},$$

for some choice of m.

Question 1.1 (6p)

Implement a genetic algorithm that takes as the input a real number x (class `numeric` in `R`) a returns the corresponding to x values of p and $\{b_j\}_{j=0}^{m}$, for m as a parameter of the algorithm. You need to choose the representation of an individual; crossover, mutation, reproduction rules; and penalty function yourself. Create a random initial population. However, if you are able to analytically find the optimal p and $\{b_j\}_{j=0}^{m}$ you may include this individual in the initial population. Provide example calls to your code for 5 values of x that are **NOT** integer numbers and are **NOT** also negative powers of 2. For the calls you may assume that $m = 10$. Present how many generations are needed to find the optimal x, and how many generations are needed for it to dominate the population. Make a plot of the best found value in each generation, and clearly indicate the true x on it.

Question 1.2 (4p)

You are now moving your numbers to a computer architecture that allows for a representation on only five bits, i.e. $m_2 = 5$. This means that you need to somehow truncate your values. One way to do this is just to chop–off the last bits, i.e.

$$x \approx \tilde{x}_2 = 2^p + \sum_{j=1}^{m_2} b_j 2^{p-j}.$$

However, one can also alternatively take a $\tilde{x}_2$ that minimizes $|\tilde{x}_2 - x|$. This will correspond to increasing b_{m_2} by 1 if $b_{m_2+1} = 1$ and to 0 otherwise. What happens to bits $m_2 - 1, m_2 - 2, \ldots, 1$ if $b_{m_2} = 1$? First implement a function that takes as its input the output of your approximation from Q1.1, and chops–off the lower bits. Then, it implements the above described improvement. Check for your choice of xs from Q1.1 that the above described improvement is better than simply chopping–off bits. If you failes in Q1.1, then choose some values of p and take some binary sequences of length 10. Provide these as input to your function. Calculate the value of x corresponding to them.

Assignment 2 (10p)

Consider the function
$$f(x) = x \sin(\frac{\pi}{2} x^2).$$

Question 2.1 (1p)

Present a plot of the function f over negative and positive numbers. Will any of the considered in our course methods actually have a chance of finding the global minimum?

Question 2.2 (4p)

Implement a simulated annealing that tries to minimize $f(\cdot)$. The temperature and acceptance probability functions should be parameters of your algorithm.

Question 2.3 (3p)

Compare how your simulated annealing algorithm behaves for different (at least two of each) choices of temperature and acceptance probability functions, and starting value. Plot the final found minima on the graph of $f(\cdot)$, clearly indicate what was the starting point.
If you failed to implement your own algorithm you may do this question using SANN method of `optim()`. You need to experiment with different control settings (argument `control`, read `?optim`) and compare how the optimization performs.

Question 2.4 (2p)

Use the `"Nelder-Mead"` method of `optim()` to minimize $f(\cdot)$. Try it out for different starting points. How does it compare with the simulated annealing algorithm? Better or worse? Where is the final found point with respect to the initial point?