Nonlinear fit of parameters of logistic function

A logistic function with parameters $a$, $b$, and $c$ is a function $f$ of a single variable given by

$$f(t) = f(t; a, b, c) = \frac{a}{1 + \left(\frac{a}{b} - 1\right)e^{-ct}} = \frac{ab}{b + (a - b)e^{-ct}}.$$  

The objective of this homework is to find optimal values of parameters $a$, $b$, and $c$, for which the values of $f$ will be the best fit for the data $(t_1, y_1), (t_2, y_2), \ldots, (t_n, y_n)$ in the sense of nonlinear least squares method. Precisely: We want to make the value of the expression

$$\sum_{i=1}^{n} (f(t_i) - y_i)^2 = (f(t_1) - y_1)^2 + (f(t_2) - y_2)^2 + \cdots + (f(t_n) - y_n)^2$$

as small as possible.

Some background: The function $f$ is a solution of the logistic differential equation

$$f''(t) = cf(t) \left(1 - \frac{f(t)}{a}\right),$$

which represents a relatively simple way of modelling population growth with limited resources. The parameter $c$ represents growth rate, parameter $a$ is the carrying capacity, parameter $b$, which is the integrating constant of the differential equation, is the initial size of the population, $b = f(0)$. The idea is that the growth of the population $f''(t)$ is proportional to the size of the population as long as this is small compared to the carrying capacity ($f(t)$ is much smaller than $a$). Once the size of the population grows closer to the carrying capacity ($f(t)$ is comparable in size to $a$) the expression on the right-hand side of the differential equation gets closer to 0 and growth slows down considerably.

**Approach: Gauss–Newton iteration**

For the data set $(t_1, y_1), \ldots, (t_n, y_n)$ define a vector-valued function

$$F: \mathbb{R}^3 \rightarrow \mathbb{R}^n, F([a, b, c]^T) := [f(t_1) - y_1, f(t_2) - y_2, \ldots, f(t_n) - y_n]^T,$$

where $f(t; a, b, c) = \frac{ab}{b + (a - b)e^{-ct}}$ is the logistic function with parameters $a$, $b$, and $c$. Concisely: the $i$th component of the function $F$ is $F_i([a, b, c]^T) = f(t_i; a, b, c) - y_i$. We'd like to find the vector of parameters $x = [a, b, c]^T$, for which the value of $\|F(x)\|^2$ is the smallest possible.
We’ll find \( \mathbf{x} \) using the Gauss–Newton iteration: In one step of this iteration the current solution approximation \( \mathbf{x}_k \) is replaced by an improved approximation
\[
\mathbf{x}_{k+1} = \mathbf{x}_k - (JF(\mathbf{x}_k))^+F(\mathbf{x}_k),
\]
where \((JF(\mathbf{x}_k))^+\) is the Moore–Penrose pseudoinverse of the Jacobi matrix of the function \(F\) at \(\mathbf{x}_k\). This is of course just a lazy, concise (and unsuitable) way of expressing one step of this method by a formula. The Moore–Penrose pseudoinverse of \(JF(\mathbf{x}_k)\) is never evaluated in practice. In fact we directly solve \((JF(\mathbf{x}_k))\mathbf{y}_k = F(\mathbf{x}_k)\) in the sense of the linear least squares method and then take \(\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{y}_k\). (That’s precisely what is achieved by using \texttt{newton.m} from problem sessions.) Moreover, the method only ‘works’ if \(JF(\mathbf{x}_k)\) has full rank. (Why?)

**Task**

1. Explicitly write down the expression for \(F = [F_1, F_2, \ldots, F_n]^T\) given the data set
\[
(t_1, y_1), \ldots, (t_n, y_n).
\]
Clearly give the expression for \(F_i([a, b, c]^T)\) in case \(f\) is the logistic function.

2. Find \(JF\). Clearly give the expression for the \(i\)th row of the Jacobi matrix of \(F\) in case \(f\) is the logistic function.

3. Write an octave function
\[
\mathbf{X} = \text{nonlinearFit}(\text{data}, \mathbf{f}, \text{pgradf}, \mathbf{X0}, \text{tol}, \text{maxit}),
\]
which given arguments
   - data set \(\text{data} = [t_1 \ y_1 \ t_2 \ y_2 \ \ldots \ t_n \ y_n]\),
   - a multivariate function \(\mathbf{f}\), which we view as a function of \(t\) and parameters \(\mathbf{x} = [a, b, c, \ldots]^T\), ie. \(f(t; \mathbf{x})\) or
   \[
   (t, [a, b, c, \ldots]^T) \rightarrow f(t; a, b, c, \ldots),
   \]
   - (transposed) gradient of the function \(\mathbf{x} \rightarrow f(t; \mathbf{x})\), \text{pgradf}, ie. the function
   \[
   (t, [a, b, c, \ldots]^T) \rightarrow \begin{bmatrix}
   \frac{\partial f}{\partial a}, \frac{\partial f}{\partial b}, \frac{\partial f}{\partial c}, \ldots
   \end{bmatrix},
   \]
• initial guess $X_0$ for the parameters of $f$ (ie. $x_0 = [a_0, b_0, c_0, \ldots] \top$),
• required accuracy $\text{tol}$, and
• the maximum allowed number of iterations $\text{maxit}$,
returns best-fit parameters $X = [a; b; c; \ldots]$ of the function $f$. Tip: Use $\text{newton.m}$ from problem sessions. Stick to specifications! Double check the dimensions (the number of rows and columns) of the arguments and return values of the above functions.

4. Write an octave function

$$X = \text{logisticFit}(\text{data}, X_0, \text{tol}, \text{maxit}),$$

which given arguments of the same form as above returns the best-fit parameters $X = [a; b; c]$ of the logistic function

$$f(t; a, b, c) = \frac{ab}{b + (a - b)e^{-ct}}.$$

Stick to specifications!

5. Use the function $\text{logisticFit}$ to monitor the number of confirmed cases of SARS-CoV-2 infections in different countries. Up-to-date data can be found on the web, data with several-days-delay is also accessible at [https://github.com/CSSEGISandData/COVID-19](https://github.com/CSSEGISandData/COVID-19). It’s likely that some (double) effort will be needed to ensure convergence of the Gauss–Newton iteration: Picking a proper initial guess and determining what part of data to ignore. For the latter is seems ignoring an early portion of the data is a good choice. For the former: Try to draw the data and some graphs of the logistic function for certain parameters $a$, $b$, and $c$, then choose those parameters which (visually) seem best suited.

**Submission**

Use the online classroom to submit the following:

1. files $\text{nonlinearFit.m}$ and $\text{logisticFit.m}$, which should be well commented and contain at least one test,

2. a report file $\text{solution.pdf}$ which contains the necessary derivations and answers to questions.

While you can discuss solutions of the problems with your colleagues, the programs and report must be your own creation. You can use all octave functions from problem sessions.