Homework 9

Implement the Newton method on page 10–17 of the slides for the problem

\[
\text{minimize } f(x) = -\sum_{i=1}^{m} \log(1 - a_i^T x) - \sum_{j=1}^{n} \log(1 - x_j^2).
\]

The variable \( x \) is an \( n \)-vector. The solution of this problem is the analytic center of the set of inequalities

\[
Ax \preceq 1, \quad -1 \preceq x \preceq 1,
\]

where \( A \) is an \( m \times n \) matrix with rows \( a_i^T \).

1. The starting point in Newton’s method must satisfy \( x^{(0)} \in \text{dom } f \). An obvious choice for this problem is \( x^{(0)} = 0 \). Use \( \epsilon = 10^{-8} \) in the stopping criterion on page 10–17, and \( \alpha = 0.01, \beta = 1/2 \) in the backtracking line search (page 10–6). In the exit condition for the line search on page 10–6, the left-hand side \( f(x + t\Delta x) \) is interpreted as \( +\infty \) if \( x + t\Delta x \not\in \text{dom } f \).

Test your code on randomly generated matrices \( A \) (for example, using the MATLAB command \( \text{A = randn}(m,n) \)), for different values of \( m \) and \( n \) (perhaps up to \( m = 2000, n = 1000 \)).

For a typical instance, plot the Newton decrement \( \lambda(x^{(k)}) \) and the step size \( t \) versus iteration number \( k \).

2. Suppose \( m \ll n \). Explain how to compute the Newton step \( \Delta x_{\text{nt}} \) in a number of operations that grows linearly in \( n \) (see page 10–30 of the slides). You do not need to implement this faster method.