1. Using a cannon positioned at the origin $(0,0,0)$ a target located on the xy-plane (e.g. at the point $T(400\,\mathrm{m}, 300\,\mathrm{m})$) has to be hit. The speed of the projectile at the moment of firing is $v_0 = 300\,\mathrm{m/s}$, the cannon can be arbitrarily rotated around the z-axis and its barrel inclination can be arbitrarily set. The projectile is affected by air drag with drag coefficient $c = 0.004$ (quadratic drag equation), additional annoyance is the wind blowing at constant velocity of $\mathbf{w} = [5, -2, 0]^\mathsf{T}\,\mathrm{m/s}$.

   Where do we point our cannon and how do we set its inclination to hit the target?

   (a) Write down the forces acting upon the projectile of mass m, position $\mathbf{x}$, and velocity $\mathbf{v} = \dot{\mathbf{x}}$. Now use the Newton's 2^{nd} law to confirm that the equation of motion for the projectile is

   $$\ddot{\mathbf{x}} = \mathbf{g} + c(\mathbf{w} - \dot{\mathbf{x}})\|\mathbf{w} - \dot{\mathbf{x}}\|.$$

   (b) Rewrite the above system of 3 DE's of order 2 as a system of 6 DE's of order 1.

   (c) Write the initial velocity of the projectile as

   $$\mathbf{v}_0 = v_0 \begin{bmatrix} \cos\phi\cos\theta \\ \sin\phi\cos\theta \\ \sin\theta \end{bmatrix},$$

   where angles ϕ and θ determine the initial direction. The projectile will land on the xy-plane at a point $[x, y, 0]^\mathsf{T}$. What we have is a vector-valued function of two variables $\mathbf{F}$ which, given a direction determined by ϕ and θ, returns a point $[x, y]^\mathsf{T} = \mathbf{F}([\phi, \theta]^\mathsf{T})$ on the plane. Write an octave function `T = izstrelek([phi; theta])` which returns the point `T = [x; y]` where the projectile hits the ground given initial direction determined by `[phi; theta]`. (Keep solving the system of DE's above numerically using the Runge–Kutta method until the projectile hits the xy-plane.)

   (d) Find a solution $[\phi, \theta]^\mathsf{T}$ of the equation $\mathbf{F}([\phi, \theta]^\mathsf{T}) = \mathbf{r}_T$ using the secant (or discretized Newton's, or Broyden's) method. Write an octave function `x = secant([x0, ..., xn], F, tol, maxit)`.

   Solving systems of nonlinear equations, part 2
   We already know how to find a solution of a system $\mathbf{F}(\mathbf{x}) = \mathbf{0}$, where $\mathbf{F}\colon \mathbb{R}^n \to \mathbb{R}^n$ is a continuously differentiable map, using the Newton's method. Along with the initial guess $\mathbf{x}^{(0)}$ and the function $\mathbf{F}$ the method also requires the Jacobi matrix $J\mathbf{F}$ of the function $\mathbf{F}$. What do we do if the Jacobi matrix is *not available*? (The Jacobi matrix may be hard to evaluate, evaluation may not be worth the effort, or the evaluation of $J\mathbf{F}$ may be too time consuming.) We'll describe three alternatives and execute one (maybe two) of these alternatives.

   - *Discretized Newton's method* replaces the partial derivatives $\partial F_i/\partial x_j$ with finite differences of the form

     $$\frac{\partial F_i}{\partial x_j}(\mathbf{x}) \doteq \frac{F_i(\mathbf{x} + h\mathbf{e}_j) - F_i(\mathbf{x})}{h},$$

where h is a well-chosen (small) number (usually $h \geq \sqrt{\epsilon}$). This is used to to build an approximation of the Jacobi matrix at each step of the iteration. The rest is the same as the usual Newton's method. A drawback: Each step requires us to evaluate $n+1$ function values (of the function $\mathbf{F}$). Advantage: The order of convergence is practically the same as the usual Newton's method, ie. 2, and *does not* depend on n (the dimension of $\mathbb{R}^n$).

- *Multidimensional secant method* is a generalization of the secant method for a function f of a single variable x and the equation $f(x) = 0$. The (1-dimensional) secant method starts with initial guesses $x^{(0)}$ and $x^{(1)}$, finds the equation $y = ax + b$ of a line passing through points $(x^{(0)}, f(x^{(0)}))$ and $(x^{(1)}, f(x^{(1)}))$, and then solves the equation $ax + b = 0$. Its solution $x^{(2)}$ is a new initial guess, $x^{(0)}$ is discarded. We use $x^{(1)}$ and $x^{(2)}$ as initial guesses for the next step of the iteration...

  The multidimensional secant method approximates the map $\mathbf{F}$ by an affine map $\mathbf{x} \mapsto A\mathbf{x} + \mathbf{b}$ (that's a direct generalization of $x \mapsto ax + b$). To do this we'll need $n+1$ initial guesses $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$. The solution $\mathbf{x}^{(n+1)}$ of the equation $A\mathbf{x} + \mathbf{b} = \mathbf{0}$ is then added to the solution guesses while $\mathbf{x}^{(0)}$ is discarded. Repeat...

  Let us describe one step of this iteration in more detail. At first glance it appears that each step requires us to solve a linear system of the form

$$\begin{aligned} A\mathbf{x}^{(0)} + \mathbf{b} &= \mathbf{F}(\mathbf{x}^{(0)}), \\ A\mathbf{x}^{(1)} + \mathbf{b} &= \mathbf{F}(\mathbf{x}^{(1)}), \\ &\vdots \\ A\mathbf{x}^{(n)} + \mathbf{b} &= \mathbf{F}(\mathbf{x}^{(n)}), \end{aligned}$$

  where the unknowns are $A = [a_{ij}]$ and $\mathbf{b} = [b_i]^\mathsf{T}$. (That is a system of $n^2 + n$ equations in $n^2 + n$ unknowns!) Once A and $\mathbf{b}$ are obtained, we solve $A\mathbf{x} = -\mathbf{b}$ to get $\mathbf{x}^{(n+1)}$. (This is a much smaller system; it has n equations in n unknowns.) However, it turns out that $\mathbf{x}^{(n+1)}$ can be obtained almost directly by solving an appropriate system of $n+1$ linear equations. Here's how: Let $X = [\mathbf{x}^{(0)}, \ldots, \mathbf{x}^{(n)}]$ be the matrix of initial approximations (that's a $n \times (n+1)$ matrix). We now get a new approximation $\mathbf{x}^{(n+1)}$ by first solving $Z\mathbf{z} = \mathbf{e}_1$, where

$$Z = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \mathbf{F}(\mathbf{x}^{(0)}) & \mathbf{F}(\mathbf{x}^{(1)}) & \cdots & \mathbf{F}(\mathbf{x}^{(n)}) \end{bmatrix}, \quad \mathbf{e}_1 = [1, 0, \ldots, 0]^\mathsf{T},$$

  and then set $\mathbf{x}^{(n+1)} = X\mathbf{z}$.

  Advantage of the secant method: Each step (except the first one) requires us to evaluate one new function value. Disadvantage: The order of convergence depends on n, it is the positive solution of the equation $t^{n+1} - t^n - 1 = 0$. (At $n = 1$ we have $t \doteq 1.618$, at $n = 2$ it is $t \doteq 1.466$, while at $n = 3$ we have $t \doteq 1.380, \ldots$)

- *The Broyden method* is similar to (discretized) Newton's method. Instead of evaluating (the approximation) of the Jacobi matrix at each step we simply update the existing (approximation of) the Jacobi matrix of $\mathbf{F}$. Let $J_0 = J\mathbf{F}(\mathbf{x}^{(0)})$ be (an approximation for) the Jacobi matrix of $\mathbf{F}$ at $\mathbf{x}^{(0)}$. Let J_k be the approximation for the Jacobi matrix $\mathbf{F}$ at kth step, and $\mathbf{x}^{(k)}$ the approximation of the iteration at the kth step. The new approximation for the iteration is then
$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - J_k^{-1}\mathbf{F}(\mathbf{x}^{(k)}).$$
We require that the new approximation J_{k+1} of the Jacobi matrix satisfies the *secant equation*
$$J_{k+1}(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = \mathbf{F}(\mathbf{x}^{(k+1)}) - \mathbf{F}(\mathbf{x}^{(k)}).$$
There are many matrices J_{k+1} satisfying this equation. Broyden's method picks
$$J_{k+1} = J_k + \frac{1}{\|\mathbf{d}_k\|^2}\Big(\mathbf{F}(\mathbf{x}^{(k+1)}) - \mathbf{F}(\mathbf{x}^{(k)}) - J_k\mathbf{d}_k\Big)\mathbf{d}_k^\top,$$
where $\mathbf{d}_k = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}$. (Verify that this J_{k+1} actually satisfies the secant equation!)

  The biggest advantage of the Broyden's method compared to discretized Newton's method is that it only requires a single function evaluation at each step. Disadvantage: More steps are required for convergence as we only approximately update the (approximation of) the Jacobi matrix at each step. This is not a big disadvantage; the steps of Broyden's method are quick to evaluate, since we don't need $n+1$ function evaluations at each step. This usually means that we obtain a solution faster.