Intersection of two implicit surfaces

A surface in $\mathbb{R}^3$ can be described as a solution set of an equation $f(\mathbf{x}) = 0$, where $\mathbf{x} = [x_1, x_2, x_3]^\mathsf{T} \in \mathbb{R}^3$, and f is a function of three variables. Suppose we are given two surfaces given by $f_1(\mathbf{x}) = 0$ and $f_2(\mathbf{x}) = 0$. The intersection of these two surfaces is the solution set of the nonlinear system

$$\begin{aligned} f_1(\mathbf{x}) &= 0, \\ f_2(\mathbf{x}) &= 0. \end{aligned}$$

If f_1 and f_2 are smooth functions and some additional conditions are satisfied the intersection of these two surfaces is a smooth curve K. The objective is to find this curve.

Construction of the curve K

View equations $f_1(\mathbf{x}) = 0$ and $f_2(\mathbf{x}) = 0$ as equations of level sets of f_1 and f_2. The curve K is the intersection of these two level sets. Gradients of f_1 and f_2 are orthogonal to K at each point of K. In other words, the vector $(\operatorname{grad} f_1) \times (\operatorname{grad} f_2)$ is tangent to K. Set

$$\mathbf{F}(\mathbf{x}) = \frac{(\operatorname{grad} f_1(\mathbf{x})) \times (\operatorname{grad} f_2(\mathbf{x}))}{\|(\operatorname{grad} f_1(\mathbf{x})) \times (\operatorname{grad} f_2(\mathbf{x}))\|}.$$

This is a vector-valued function $\mathbf{F}\colon \mathbb{R}^3 \to \mathbb{R}^3$.

Suppose we start at $\mathbf{x}_0 \in K$, ie. $f_1(\mathbf{x}) = 0$ and $f_2(\mathbf{x}) = 0$. Take a step of length h (for some small $h > 0$) out of $\mathbf{x}_0$ in the direction tangent to K to get a new point $\mathbf{y}$. According to the definition of $\mathbf{F}$, that means $\mathbf{y} = \mathbf{x}_0 + h\mathbf{F}(\mathbf{x}_0)$, since $\|\mathbf{F}(\mathbf{x})\| = 1$. This $\mathbf{y}$ does *not* (necessarily) lie on the curve K, but it is sensible to assume that it si close to K (since h is small). Denoting $\mathbf{v} = \mathbf{F}(\mathbf{y})$, $\mathbf{v} \cdot \mathbf{x} = \mathbf{v} \cdot \mathbf{y}$ is the equation of a plane, which is 'close' to a plane normal to K. We can use $\mathbf{y}$ to obtain $\mathbf{x}_1$, which actually lies on K by solving the system of nonlinear equations (in the unknown $\mathbf{x}$)

$$\begin{aligned} f_1(\mathbf{x}) &= 0, \\ f_2(\mathbf{x}) &= 0, \\ \mathbf{v} \cdot \mathbf{x} - \mathbf{v} \cdot \mathbf{y} &= 0. \end{aligned} \tag{1}$$

We will obtain the solution $\mathbf{x}_1$ of this system by using the Newton's iteration with initial guess $\mathbf{y}$. Expectation is, of course, that $\mathbf{x}_1$ is close $\mathbf{y}$. (If it is not, our choice of h was too large.)

Let's summarize the construction of consecutive points on the curve K:

(i) Take a step of length h out of $\mathbf{x}_0 \in K$ in the direction of $\mathbf{F}(\mathbf{x}_0)$ to get an intermediate approximation $\mathbf{y}$,

(ii) then use the system (1) to find a next point $\mathbf{x}_1 \in K$ on the curve.

Repeat this for $\mathbf{x}_1$ and (again through the intermediate approximation and system (1)) get $\mathbf{x}_2 \in K$, etc. What we get is a sequence of points

$$\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n,$$

describing the curve K.

A minor problem regarding the sensibility of this construction: In practice the starting point $\mathbf{x}_0 \in K$ is not known, we only know an approximation $\mathbf{y}$ for $\mathbf{x}_0$. A quick remedy: Solve the system (1) with this approximation, get $\mathbf{x}_0 \in K$, and use the method described above.

Task

1. Write down the vector-valued function $\mathbf{G}\colon \mathbb{R}^3 \to \mathbb{R}^3$ and its Jacobi matrix $J\mathbf{G}$ corresponding to the system (1). (Both can be expressed using f_i, grad f_i, $\mathbf{y}$, and $\mathbf{v}$.)

2. Write a Julia function

   ```
   X = presekPloskev(f1, gradf1, f2, gradf2, X0, h, n, tol, maxit),
   ```

   which given:

   - functions $f_1, f_2\colon \mathbb{R}^3 \to \mathbb{R}$, functions of a vector argument $\mathbf{x} \in \mathbb{R}^3$,
   - gradients grad f_1, grad $f_2\colon \mathbb{R}^3 \to \mathbb{R}^3$, vector-valued functions $\mathbb{R}^3 \to \mathbb{R}^3$,
   - approximation `X0` for the initial point on the curve (this has to be 'adjusted' using the system (1) first),
   - step length `h`,
   - the number `n` of consecutive points on the curve to be constructed,
   - the tolerance `tol` for Newton's iteration, and
   - maximum allowed number of iteration steps `maxit` for Newton's iteration

   returns a Vector of size $n+1$ containing Tuples of size 3 representing the points on the curve K. *Stick to specifications!*

Submission

Use the online classroom to submit the following:

1. file **presekPloskev.jl**, which should be well commented and contain at least one test,

2. a report file **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 Julia functions from problem sessions.

Extra credit task 

The method described is in fact Euler method for solving systems of ordinary differential equations. Indeed, the natural parametrization $\mathbf{x}(t)$ of K is a solution of a system of autonomous differential equations

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}).$$

Picking $\mathbf{x}(0) = \mathbf{x}_0$ as our initial condition, where $\mathbf{x}_0 \in K$, the solution is precisely the natural parametrization of K.

Use one step of the Runge–Kutta method of 4^{th} order with step size $h > 0$ to determine the intermediate approximation $\mathbf{y}$, which you use along system (1) to find the next point on the curve K. Write a Julia function

```
X = presekPloskevRK4(f1, gradf1, f2, gradf2, X0, h, n, tol, maxit),
```

which accepts the same arguments and returns the same type values as the function `presekPloskev`, but uses the Runge–Kutta method of 4^{th} order (instead of the Euler method) to find intermediate approximations. Compare the performance of both functions.