Nonlinear Least Squares

1. Gauss-Newton Method
   - nonlinear overdetermined systems
   - near intersection points of three circles
   - a computation with Julia
2. Singular Value Decomposition
   - yet another matrix factorization ...
   - computing the condition number of a matrix
3. Levenberg-Marquardt Method
   - adding a regularization term
   - description of the method

MCS 471 Lecture 21
Numerical Analysis
Jan Verschelde, 10 October 2022

Nonlinear Least Squares

1. Gauss-Newton Method
   - nonlinear overdetermined systems
   - near intersection points of three circles
   - a computation with Julia
2. Singular Value Decomposition
   - yet another matrix factorization ...
   - computing the condition number of a matrix
3. Levenberg-Marquardt Method
   - adding a regularization term
   - description of the method

nonlinear overdetermined systems

Let $\mathbf{f} = (f_1, f_2, \ldots, f_m)$ define the overdetermined system $\mathbf{f}(\mathbf{x}) = \mathbf{0}$, with

1. m equations $f_i(\mathbf{x}) = 0$, $i = 1, 2, \ldots, m$,
2. in n variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)$.

where $m > n$.

Solving $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ in the least squares sense
means finding a solution $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ such that

$$\|\mathbf{f}(\mathbf{z})\|_2^2 = f_1(\mathbf{z})^2 + f_2(\mathbf{z})^2 + \cdots + f_m(\mathbf{z})^2$$

is minimal. $\|\mathbf{f}(\mathbf{z})\|_2$ is the norm of the residual, or backward error.

There may be many solutions of $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.

the Jacobian matrix

We need the matrix of all first order partial derivatives

$$J_{\mathbf{f}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}.$$

$J_{\mathbf{f}}$ is the Jacobian matrix of the system $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.

For a system $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ of m equations in n variables, $J_{\mathbf{f}}$ is an m-by-n matrix.

one Gauss-Newton step

Given are $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ and $\mathbf{x}_0 \in \mathbb{R}^n$.
The $\mathbf{x}_0$ contains approximations for the coordinates of a solution.

1. Evaluate $\mathbf{f}$ at $\mathbf{x}_0$, $\mathbf{b} := -\mathbf{f}(\mathbf{x}_0)$.
2. Evaluate $J_{\mathbf{f}}$ at $\mathbf{x}_0$, $A := J_{\mathbf{f}}(\mathbf{x}_0)$.
3. Solve $A\Delta\mathbf{x} = \mathbf{b}$.
4. Update $\mathbf{x}_0$: $\mathbf{x}_1 := \mathbf{x}_0 + \Delta\mathbf{x}$.

Consider the linear system $A\Delta\mathbf{x} = \mathbf{b}$:

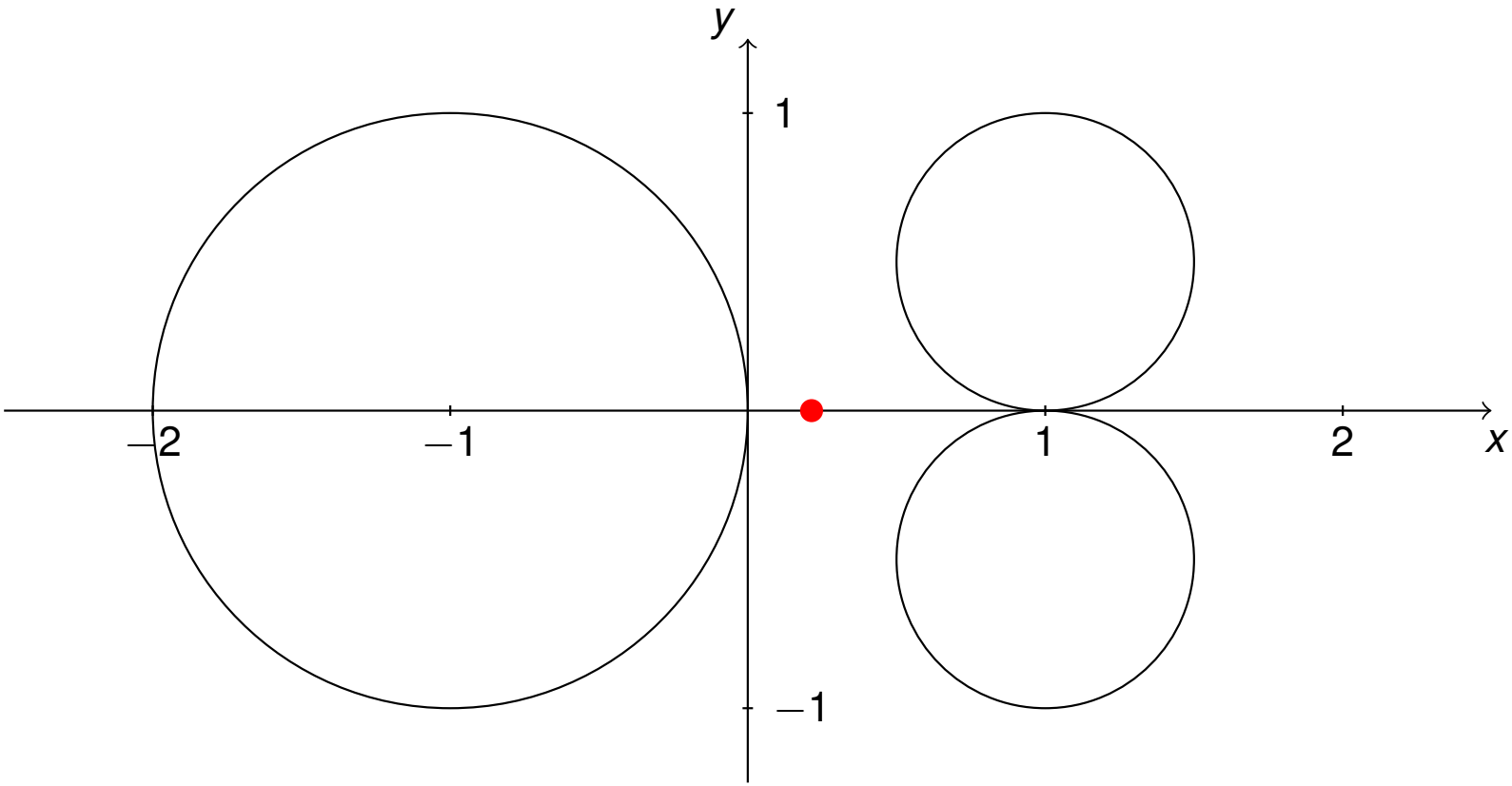
- As $\mathbf{f}$ defines m equations, $\mathbf{b} \in \mathbb{R}^m$.
- Since $J_{\mathbf{f}}$ is an m-by-n matrix, with $m > n$, the matrix A is also m-by-n.

We solve $A\Delta\mathbf{x} = \mathbf{b}$ in the least squares sense, preferably with a QR factorization.

Nonlinear Least Squares

1. Gauss-Newton Method
   - nonlinear overdetermined systems
   - near intersection points of three circles
   - a computation with Julia
2. Singular Value Decomposition
   - yet another matrix factorization ...
   - computing the condition number of a matrix
3. Levenberg-Marquardt Method
   - adding a regularization term
   - description of the method

near intersection points of three circles



$$\text{minimize} \sum_{k=1}^{3} \left(\sqrt{(x - x_k)^2 + (y - y_k)^2} - r_k \right)^2$$

a system of nonlinear equations

$$\text{minimize} \sum_{k=1}^{3} \left(\sqrt{(x-x_k)^2 + (y-y_k)^2} - r_k \right)^2$$

for $(x_1, y_1) = (-1, 0)$, $r_1 = 1$,
$(x_2, y_2) = (1, 0.5)$, $r_2 = 0.5$, and
$(x_3, y_3) = (1, -0.5)$, $r_3 = 0.5$.

We apply the method of Gauss-Newton to

$$\begin{cases} \sqrt{(x+1)^2 + y^2} - 1 &= 0 \\ \sqrt{(x-1)^2 + (y-0.5)^2} - 0.5 &= 0 \\ \sqrt{(x-1)^2 + (y+0.5)^2} - 0.5 &= 0. \end{cases}$$

Nonlinear Least Squares

1. Gauss-Newton Method
   - nonlinear overdetermined systems
   - near intersection points of three circles
   - a computation with Julia
2. Singular Value Decomposition
   - yet another matrix factorization ...
   - computing the condition number of a matrix
3. Levenberg-Marquardt Method
   - adding a regularization term
   - description of the method

a computation with Julia

Recall the script `multinewton.jl` from Lecture L-13.

In the function `main()`, we define the equations as

```
one = "sqrt((x+1)*(x+1) + y*y) - 1"
two = "sqrt((x-1)*(x-1) + (y-0.5)*(y-0.5)) - 0.5"
three = "sqrt((x-1)*(x-1) + (y+0.5)*(y+0.5)) - 0.5"
```

The other adjustments consists in adding a third `h`
next to the `f` and `g` equations.

the same function `NewtonStep`

```
"""
Does one step of the Gauss-Newton method,
on the vector function fun and Jacobian jac,
starting at x0, y0.  Returns the new values
for x and y, the norm of the update, and
the norm of the residual at x0, y0.
"""
function NewtonStep(fun, jac,
                    x0::Float64, y0::Float64)
    valfun = -SymPyFun(fun, x0, y0)
    nfx = norm(valfun)
    valmat = SymPyMatrixEvaluate(jac, x0, y0)
    update = valmat\valfun
    ndx = norm(update)
    x1 = x0 + update[1]
    y1 = y0 + update[2]
    return [x1, y1, ndx, nfx]
end
```

running `NewtonStep`

Starting at $(0, 0)$, we print after each step

1. x, the value of the first coordinate of the solution
2. y, the value of the second coordinate of the solution
3. $\|(\Delta x, \Delta y)\|_2$, the norm of the update, estimates the forward error
4. $\|\mathbf{f}(x, y)\|_2$, the norm of the residual, estimates the backward error

```
step                x                               y                  |update|    |f(x,y)|
  0 :  0.0000000000000000e+00   0.0000000000000000e+00
  1 :  4.2522031115387854e-01   8.7770836714417518e-17  4.25e-01    8.74e-01
  2 :  4.1112530855930013e-01  -9.1648644610152970e-17  1.41e-02    5.64e-01
  3 :  4.1313149307582292e-01   1.5093412638940774e-16  2.01e-03    5.64e-01
  4 :  4.1285833541953565e-01  -3.0643041052992871e-17  2.73e-04    5.64e-01
  5 :  4.1289576405778744e-01   9.0441053455228248e-17  3.74e-05    5.64e-01
  6 :  4.1289063992215830e-01  -9.1178377175993533e-17  5.12e-06    5.64e-01
  7 :  4.1289134152035289e-01   2.1152220516108437e-16  7.02e-07    5.64e-01
  8 :  4.1289124545886885e-01  -1.5171824204619955e-16  9.61e-08    5.64e-01
  9 :  4.1289125861145282e-01   1.5098215933337173e-16  1.32e-08    5.64e-01
 10 :  4.1289125681062272e-01  -3.0638079136074756e-17  1.80e-09    5.64e-01
```

Why so slow?

convergence of the Gauss-Newton method

With the backslash operator, Gauss-Newton is the same as Newton.

- The initial point must be close enough to the solution.
- If the solution makes all equations zero,
  then we may expect quadratic convergence.

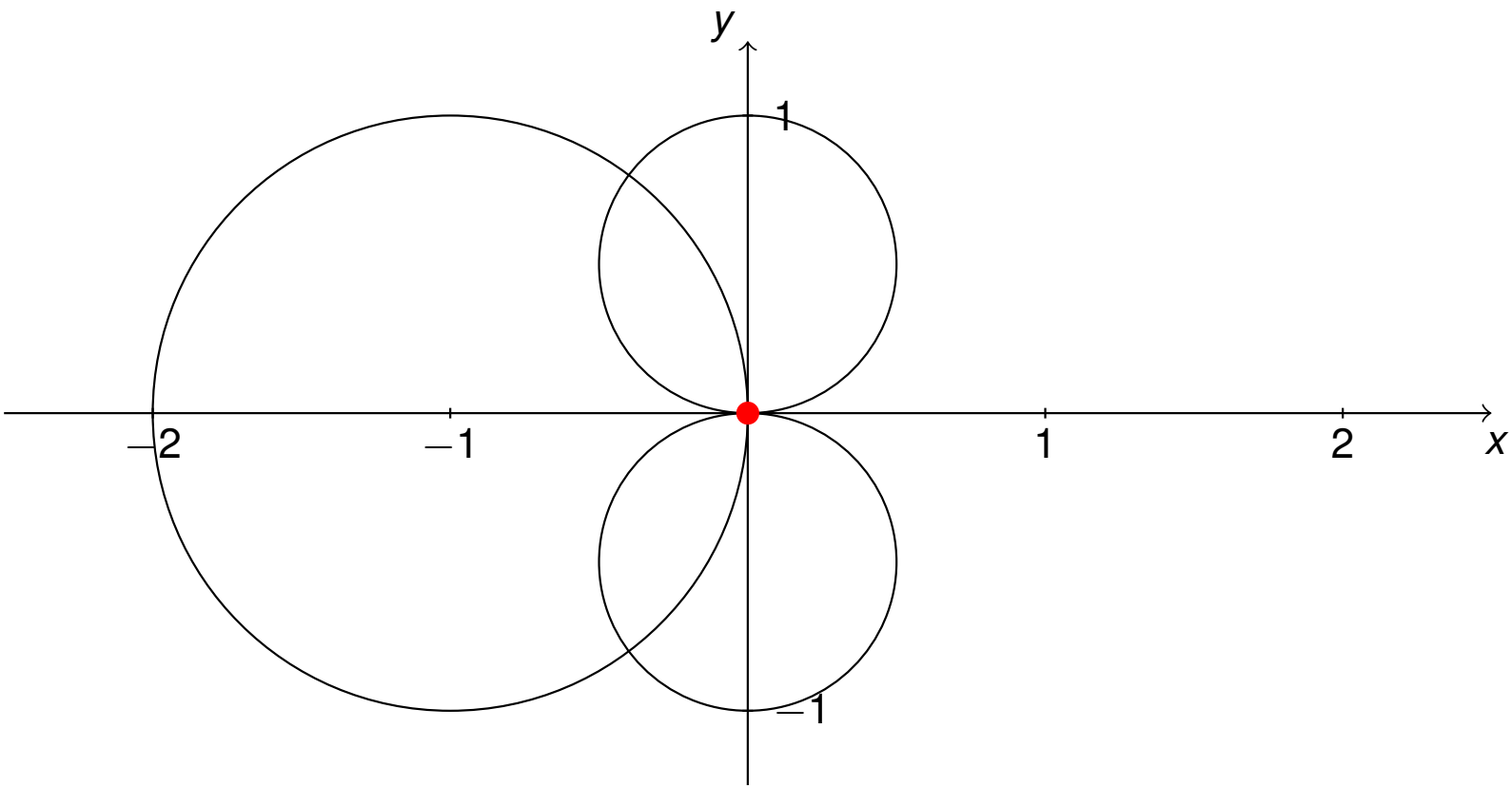
Gauss-Newton differs from Newton

- if there is *no* solution that makes all equations zero, or
- if there is only a least squares solution.

If the overconstrained problem is then *well posed*,
then we may expect local, linear convergence.

the intersection point of three circles

Exercise 1: Consider the modification of the three circle problem:



The centers of the two smaller circles are at $(0, \pm 0.5)$.
Run the method of Gauss-Newton, starting at $(0.5, 0.0)$.
Is the convergence quadratic?

Nonlinear Least Squares

1. Gauss-Newton Method
   - nonlinear overdetermined systems
   - near intersection points of three circles
   - a computation with Julia
2. Singular Value Decomposition
   - yet another matrix factorization ...
   - computing the condition number of a matrix
3. Levenberg-Marquardt Method
   - adding a regularization term
   - description of the method

the Singular Value Decomposition

Let A by an n-by-k matrix, $k \leq n$.

The Singular Value Decomposition (SVD) of A is defined by

1. U, an n-by-n orthogonal matrix, $U^T U = I$,
2. S, a diagonal n-by-k matrix with the singular values, and
3. V, is an k-by-k orthogonal matrix, $V^T V = I$,

such that $A = USV^T$ or equivalently: $U^T AV = S$.

Two important properties of the SVD.

1. The SVD provides a *diagonalization of A*, it generalizes QR.
2. The singular values are sorted in descending order.
   Given a tolerance ϵ, *the numerical rank of A*
   equals the number of singular values larger than ϵ.

the `svd` in a Julia session

```
julia> using LinearAlgebra

julia> A = rand(3, 2)
3×2 Array{Float64,2}:
 0.887977  0.173145
 0.768222  0.971541
 0.346458  0.481141

julia> U, S, V = svd(A)
SVD{Float64,Float64,Array{Float64,2}}
U factor:
3×2 Array{Float64,2}:
 -0.502462   0.863655
 -0.781218  -0.433491
 -0.370447  -0.257265
singular values:
2-element Array{Float64,1}:
 1.558484008477351
 0.5245922051038863
Vt factor:
2×2 Array{Float64,2}:
 -0.753724  -0.657191
  0.657191  -0.753724
```

verification of the properties

The session with Julia continues:

```
julia> A
3×2 Array{Float64,2}:
 0.887977  0.173145
 0.768222  0.971541
 0.346458  0.481141

julia> U*diagm(S)*transpose(V)
3×2 Array{Float64,2}:
 0.887977  0.173145
 0.768222  0.971541
 0.346458  0.481141

julia> transpose(U)*A*V
2×2 Array{Float64,2}:
 1.55848       4.44089e-16
 3.88578e-16   0.524592
```

Indeed, $USV^T = A$ and $U^T AV = S$.

last output of the three circle problem

After 10 iterations with Gauss-Newton on the problem of intersecting three circles, we end with $\|\Delta x\| = 1.80 \cdot 10^{-9}$.

Evaluating the solution then at the Jacobian matrix and computing the singular values shows:

```
The Jacobian matrix :
3x2 Array{Float64,2}:
   1.000e+00  -2.168e-17
  -7.613e-01  -6.484e-01
  -7.613e-01   6.484e-01
The singular values of the Jacobian matrix :
2-element Array{Float64,1}:
 1.469e+00
 9.169e-01
```

We see that the columns of the Jacobian matrix are linearly independent, which shows in the two nonzero singular values. So the problem of intersecting three circles is well posed.

Nonlinear Least Squares

1. Gauss-Newton Method
   - nonlinear overdetermined systems
   - near intersection points of three circles
   - a computation with Julia
2. Singular Value Decomposition
   - yet another matrix factorization ...
   - computing the condition number of a matrix
3. Levenberg-Marquardt Method
   - adding a regularization term
   - description of the method

computing the condition number of a matrix

For square matrices A, we derived $\kappa(A) = \|A^{-1}\|\|A\|$ as the condition number of the matrix A.

But what if A is not square and A^{-1} does not exist?

Let A by an n-by-k matrix, $k \leq n$, and $A = USV^T$.

The k singular values of A are $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k$.

Then the condition number $\kappa(A)$ of A is

$$\kappa(A) = \frac{\sigma_1}{\sigma_k},$$

provided $\sigma_k \neq 0$.

verification in a Julia session

```
julia> A = rand(3,3)
3×3 Array{Float64,2}:
 0.296377  0.311904   0.777915
 0.648831  0.316846   0.207123
 0.485538  0.0975773  0.837667

julia> cond(A)
7.296520716757458

julia> U, S, V = svd(A);

julia> S
3-element Array{Float64,1}:
 1.4224222554304933
 0.4901166427005800 3
 0.1949452774339017

julia> S[1]/S[3]
7.296520716757455
```

solving linear systems in the least squares sense

If $A = USV^T$, then $A\mathbf{x} = \mathbf{b}$ turns into $USV^T\mathbf{x} = \mathbf{b}$, or $S\mathbf{y} = U^T\mathbf{b}$, with $\mathbf{y} = V^T\mathbf{x}$ or $\mathbf{x} = V\mathbf{y}$.

The above formulas can be used to solve $A\mathbf{x} = \mathbf{b}$.

Exercise 2:

Consider the system $A\mathbf{x} = \mathbf{b}$ with $A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{bmatrix}$, and $\mathbf{b} = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}$.

Compute the singular value decomposition of A and use it to solve $A\mathbf{x} = \mathbf{b}$.

Compare your result with the output of $\mathbf{x} = A\backslash\mathbf{b}$.

Nonlinear Least Squares

1. Gauss-Newton Method
   - nonlinear overdetermined systems
   - near intersection points of three circles
   - a computation with Julia
2. Singular Value Decomposition
   - yet another matrix factorization ...
   - computing the condition number of a matrix
3. Levenberg-Marquardt Method
   - adding a regularization term
   - description of the method

adding a regularization term

To remedy the conditioning of the normal equations, we use a regularization parameter λ.

Let $A = J_{\mathbf{f}}(\mathbf{x}_0)$, instead of $A\Delta\mathbf{x} = -\mathbf{f}(\mathbf{x}_0)$, consider

$$\left(A^T A + \lambda \text{diag}(A^T A)\right)\mathbf{v} = -A^T \mathbf{f}(\mathbf{x}_0)$$

followed by $\mathbf{x}_1 := \mathbf{x}_0 + \mathbf{v}$.

Two extreme cases are interesting:

1. $\lambda = 0$: Gauss-Newton,
2. $\lambda = \infty$: method of steepest descent.

the condition of $A^T A + \lambda \text{diag}(A^T A)$

Exercise 3:
Compare the condition number of $A^T A$
with the condition number of $A^T A + \lambda \text{diag}(A^T A)$,
for random square matrices A of prescribed condition number 10^8.

Experiment with values $\lambda = 1, 10^{-2}, 10^{-4}, 10^{-6}$
and report the observed condition numbers of $A^T A + \lambda \text{diag}(A^T A)$.

Based on your experiments, could you describe the relationship between $\kappa(A^T A)$ and $\kappa(A^T A + \lambda \text{diag}(A^T A))$?

the gradient vector

Note that we want to minimize the square of the residual:

$$r(x,y) = f_1(x,y)^2 + f_2(x,y)^2 + f_3(x,y)^2.$$

The critical points are solutions of $\frac{\partial r}{\partial x} = 0$ and $\frac{\partial r}{\partial y} = 0$.

$$\begin{aligned}
\frac{\partial r}{\partial x} &= 2f_1(x,y)\frac{\partial f_1}{\partial x} + 2f_2(x,y)\frac{\partial f_2}{\partial x} + 2f_3(x,y)\frac{\partial f_3}{\partial x} \\
&= 2\begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_2}{\partial x} & \frac{\partial f_3}{\partial x} \end{bmatrix}^T \begin{bmatrix} f_1(x,y) \\ f_2(x,y) \\ f_3(x,y) \end{bmatrix}, \text{ and similarly:} \\
\frac{\partial r}{\partial y} &= 2\begin{bmatrix} \frac{\partial f_1}{\partial y} & \frac{\partial f_2}{\partial y} & \frac{\partial f_3}{\partial y} \end{bmatrix}^T \begin{bmatrix} f_1(x,y) \\ f_2(x,y) \\ f_3(x,y) \end{bmatrix}.
\end{aligned}$$

computing the gradient vector

We may divide by two as $\frac{\partial r}{\partial x} = 0$ and $\frac{\partial r}{\partial y} = 0$.

$$\frac{\partial r}{\partial x} = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_2}{\partial x} & \frac{\partial f_3}{\partial x} \end{bmatrix}^T \begin{bmatrix} f_1(x,y) \\ f_2(x,y) \\ f_3(x,y) \end{bmatrix}$$

$$\frac{\partial r}{\partial y} = \begin{bmatrix} \frac{\partial f_1}{\partial y} & \frac{\partial f_2}{\partial y} & \frac{\partial f_3}{\partial y} \end{bmatrix}^T \begin{bmatrix} f_1(x,y) \\ f_2(x,y) \\ f_3(x,y) \end{bmatrix}$$

Then the gradient ∇r is

$$\nabla r = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_2}{\partial x} & \frac{\partial f_3}{\partial x} \\ \frac{\partial f_1}{\partial y} & \frac{\partial f_2}{\partial y} & \frac{\partial f_3}{\partial y} \end{bmatrix} \begin{bmatrix} f_1(x,y) \\ f_2(x,y) \\ f_3(x,y) \end{bmatrix} = J_{\mathbf{f}}(x,y)^T \mathbf{f}(x,y).$$

Nonlinear Least Squares

1. Gauss-Newton Method
   - nonlinear overdetermined systems
   - near intersection points of three circles
   - a computation with Julia
2. Singular Value Decomposition
   - yet another matrix factorization ...
   - computing the condition number of a matrix
3. Levenberg-Marquardt Method
   - adding a regularization term
   - description of the method

description of the method

Input: $\mathbf{f}(\mathbf{x}) = \mathbf{0}$, a system of m equations in n unknowns,
$\mathbf{x}_0$, an initial guess for a solution,
λ, value for the regularization parameter,
N, maximum number of iterations,
δ, tolerance on $||\Delta\mathbf{x}||$,
ϵ, tolerance on $||\mathbf{f}(\mathbf{x})||$.

for $k = 0, 1, \ldots, N$ do
  $\mathbf{r} := \mathbf{f}(\mathbf{x}_k)$
  $A := J_{\mathbf{f}}(\mathbf{x}_k)$
  solve $\left(A^T A + \lambda \text{diag}(A^T A)\right) \Delta\mathbf{x} = -A^T \mathbf{r}$
  $\mathbf{x}_{k+1} := \mathbf{x}_k + \Delta\mathbf{x}$
  exit when $(||\mathbf{r}|| \leq \epsilon)$ or $(||\Delta\mathbf{x}|| \leq \delta)$

define a Julia function

Exercise 4:

Use the pseudo code on the previous slide to define a function.

1. Document all parameters in `LevenbergMarquardt`, a Julia function for the method of Levenberg-Marquardt. Use good default values for N, δ, and ϵ.
2. Write the definition of your `LevenbergMarquardt`.
3. Illustrate the correctness of your function on the example of the near intersection points of three circles, using the default values for N, δ, and ϵ.
4. Experiment with $\lambda = 0, 10^{-2}, 1, 10^2, \ldots$. What is the best value for λ for the three circle problem?

a summary on least squares solving

What we covered in four lectures:

1. Solving a linear system in the least squares sense corresponds to constructing an orthogonal basis of the column space.

2. The classical Gram-Schmidt method loses orthogonality, the modified Gram-Schmidt method is numerically stable.

   The Householder QR method applies orthogonal transformations and is recommended for ill-conditioned inputs.

3. The Generalized Minimum Residual Method proceeds iteratively, solving smaller dimensional least squares problems in each step.

4. The method of Gauss-Newton updates the current approximation via the solution of a least squares system.

   Levenberg-Marquardt adds steepest descent to Gauss-Newton.