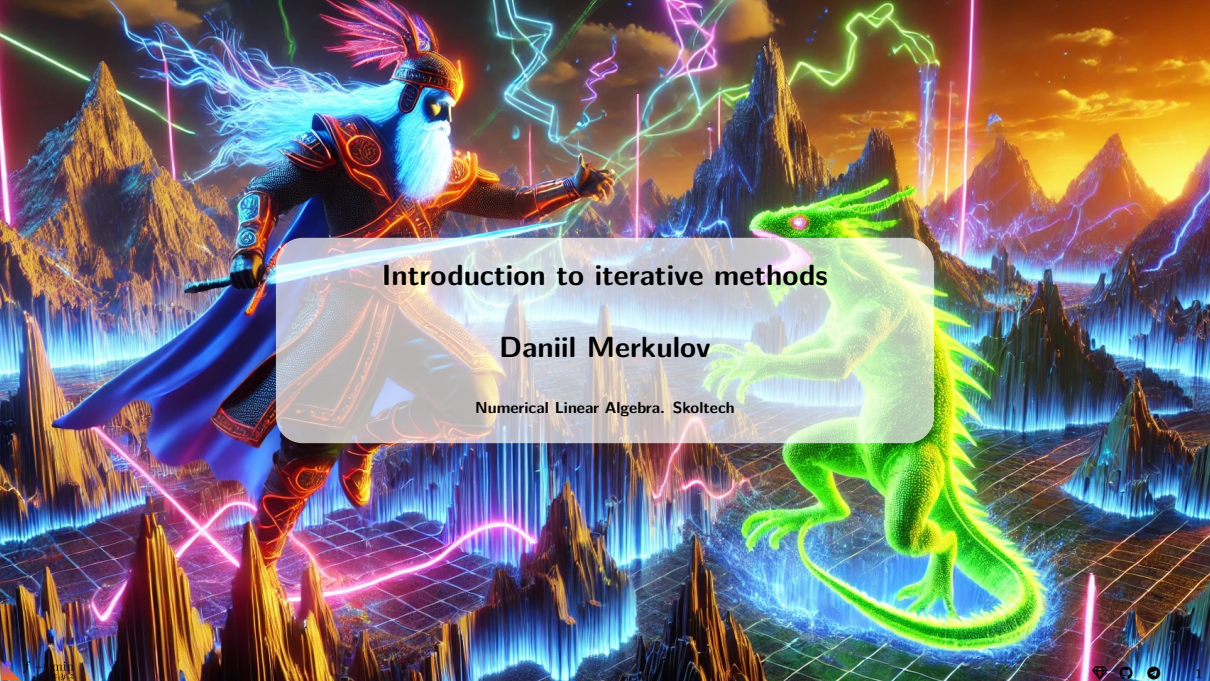


Introduction to iterative methods

Daniil Merkulov

Numerical Linear Algebra. Skoltech

Iterative methods

Iterative methods

- If we want to achieve $\mathcal{O}(N)$ complexity of solving sparse linear systems, then direct solvers are not appropriate.

Iterative methods

- If we want to achieve $\mathcal{O}(N)$ complexity of solving sparse linear systems, then direct solvers are not appropriate.
- If we want to solve partial eigenproblem, the full eigendecomposition is too costly.

Iterative methods

- If we want to achieve $\mathcal{O}(N)$ complexity of solving sparse linear systems, then direct solvers are not appropriate.
- If we want to solve partial eigenproblem, the full eigendecomposition is too costly.
- For both problems we will use iterative, Krylov subspace solvers, which treat the matrix as a **black-box** linear operator.

Matrix as a black box

- We have now an absolutely different view on a matrix: matrix is now a **linear operator**, that acts on a vector, and this action can be computed in $\mathcal{O}(N)$ operations.

Matrix as a black box

- We have now an absolutely different view on a matrix: matrix is now a **linear operator**, that acts on a vector, and this action can be computed in $\mathcal{O}(N)$ operations.
- **This is the only information** we know about the matrix: the matrix-by-vector product (matvec)

Matrix as a black box

- We have now an absolutely different view on a matrix: matrix is now a **linear operator**, that acts on a vector, and this action can be computed in $\mathcal{O}(N)$ operations.
- **This is the only information** we know about the matrix: the matrix-by-vector product (matvec)
- Can we solve linear systems using only matvecs?

Matrix as a black box

- We have now an absolutely different view on a matrix: matrix is now a **linear operator**, that acts on a vector, and this action can be computed in $\mathcal{O}(N)$ operations.
- **This is the only information** we know about the matrix: the matrix-by-vector product (matvec)
- Can we solve linear systems using only matvecs?
- Of course, we can multiply by the colums of the identity matrix, and recover the full matrix, but it is not what we need.

~~Gradient Descent~~ Richardson iteration

Richardson iteration

The simplest idea is the **“simple iteration method”** or **Richardson iteration**.

$$Ax = f,$$

Richardson iteration

The simplest idea is the **“simple iteration method”** or **Richardson iteration**.

$$Ax = f,$$

$$\tau(Ax - f) = 0,$$

Richardson iteration

The simplest idea is the **“simple iteration method”** or **Richardson iteration**.

$$Ax = f,$$

$$\tau(Ax - f) = 0,$$

$$x - \tau(Ax - f) = x,$$

Richardson iteration

The simplest idea is the **"simple iteration method"** or **Richardson iteration**.

$$Ax = f,$$

$$\tau(Ax - f) = 0,$$

$$x - \tau(Ax - f) = x,$$

$$x_{k+1} = x_k - \tau(Ax_k - f),$$

where τ is the **iteration parameter**, which can be always chosen such that the method **converges**.

Connection to ODEs

- The Richardson iteration has a deep connection to the Ordinary Differential Equations (ODE).

Connection to ODEs

- The Richardson iteration has a deep connection to the Ordinary Differential Equations (ODE).
- Consider a time-dependent problem

$$\frac{dy}{dt} + Ay = f, \quad y(0) = y_0.$$

Connection to ODEs

- The Richardson iteration has a deep connection to the Ordinary Differential Equations (ODE).
- Consider a time-dependent problem

$$\frac{dy}{dt} + Ay = f, \quad y(0) = y_0.$$

- Then $y(t) \to A^{-1}f$ as $t \to \infty$, and the **Euler scheme** reads

$$\frac{y_{k+1} - y_k}{\tau} = -Ay_k + f.$$

which leads to the Richardson iteration

$$y_{k+1} = y_k - \tau(Ay_k - f)$$

Gradient flow ODE

Let's consider the following ODE, which is referred to as the Gradient Flow equation.

$$\frac{dx}{dt} = -f'(x(t)) \tag{GF}$$

Gradient flow ODE

Let's consider the following ODE, which is referred to as the Gradient Flow equation.

$$\frac{dx}{dt} = -f'(x(t)) \qquad \text{(GF)}$$

and discretize it on a uniform grid with α step:

$$\frac{x_{k+1} - x_k}{\alpha} = -f'(x_k),$$

Gradient flow ODE

Let's consider the following ODE, which is referred to as the Gradient Flow equation.

$$\frac{dx}{dt} = -f'(x(t)) \tag{GF}$$

and discretize it on a uniform grid with α step:

$$\frac{x_{k+1} - x_k}{\alpha} = -f'(x_k),$$

where $x_k \equiv x(t_k)$ and $\alpha = t_{k+1} - t_k$ - is the grid step.
From here we get the expression for x_{k+1}

$$x_{k+1} = x_k - \alpha f'(x_k),$$

which is exactly gradient descent.
Open In Colab ♣

Gradient flow ODE

Let's consider the following ODE, which is referred to as the Gradient Flow equation.

$$\frac{dx}{dt} = -f'(x(t)) \tag{GF}$$

and discretize it on a uniform grid with α step:

$$\frac{x_{k+1} - x_k}{\alpha} = -f'(x_k),$$

where $x_k \equiv x(t_k)$ and $\alpha = t_{k+1} - t_k$ - is the grid step.
From here we get the expression for x_{k+1}

$$x_{k+1} = x_k - \alpha f'(x_k),$$

which is exactly gradient descent.
Open In Colab ♣

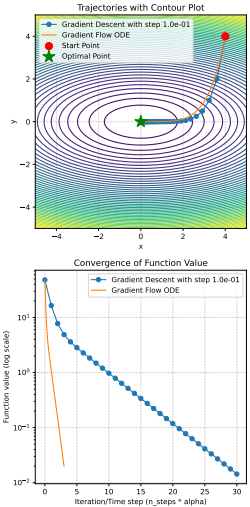


Figure 1: Gradient flow trajectory

Convergence of the Richardson method

- Let x_* be the solution; introduce an error $e_k = x_k - x_*$, then

$$e_{k+1} = (I - \tau A)e_k,$$

therefore if $\|I - \tau A\| < 1$ in any norm, the iteration converges.

Convergence of the Richardson method

- Let x_* be the solution; introduce an error $e_k = x_k - x_*$, then

$$e_{k+1} = (I - \tau A)e_k,$$

  therefore if $\|I - \tau A\| < 1$ in any norm, the iteration converges.

- For symmetric positive definite case it is always possible to select τ such that the method converges.

Convergence of the Richardson method

- Let x_* be the solution; introduce an error $e_k = x_k - x_*$, then

$$e_{k+1} = (I - \tau A)e_k,$$

  therefore if $\|I - \tau A\| < 1$ in any norm, the iteration converges.
- For symmetric positive definite case it is always possible to select τ such that the method converges.
- What about the non-symmetric case? Below demo will be presented...

Richardson method is equivalent to the Gradient Descent for quadratics

Consider the following quadratic optimization problem:

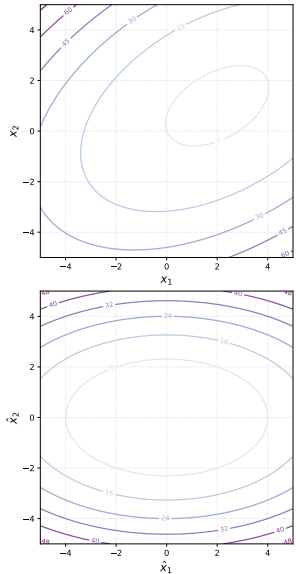
$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}^d_{++}.$$

Richardson method is equivalent to the Gradient Descent for quadratics

Consider the following quadratic optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}^d_{++}.$$

- Firstly, without loss of generality we can set $c = 0$, which will not affect optimization process.



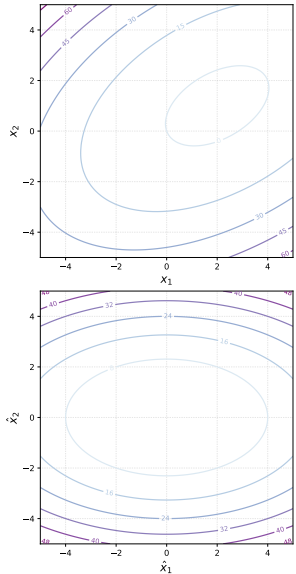
Richardson method is equivalent to the Gradient Descent for quadratics

Consider the following quadratic optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}^d_{++}.$$

- Firstly, without loss of generality we can set $c = 0$, which will not affect optimization process.
- Secondly, we have a spectral decomposition of the matrix A:

$$A = Q \Lambda Q^T$$



Richardson method is equivalent to the Gradient Descent for quadratics

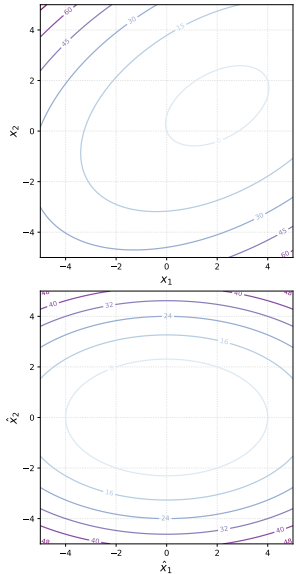
Consider the following quadratic optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}^d_{++}.$$

- Firstly, without loss of generality we can set $c = 0$, which will not affect optimization process.
- Secondly, we have a spectral decomposition of the matrix A:

$$A = Q \Lambda Q^T$$

- Let's show, that we can switch coordinates to make an analysis a little bit easier. Let $\hat{x} = Q^T(x - x^*)$, where x^* is the minimum point of initial function, defined by $Ax^* = b$. At the same time $x = Q\hat{x} + x^*$.



Richardson method is equivalent to the Gradient Descent for quadratics

Consider the following quadratic optimization problem:

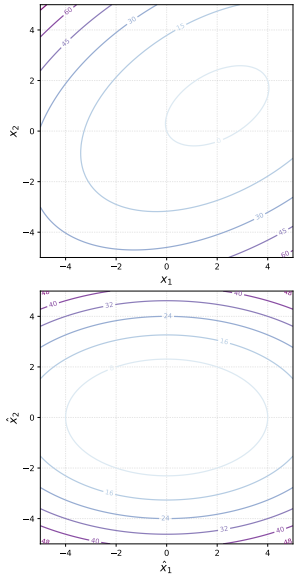
$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}^d_{++}.$$

- Firstly, without loss of generality we can set $c = 0$, which will not affect optimization process.
- Secondly, we have a spectral decomposition of the matrix A:

$$A = Q \Lambda Q^T$$

- Let's show, that we can switch coordinates to make an analysis a little bit easier. Let $\hat{x} = Q^T(x - x^*)$, where x^* is the minimum point of initial function, defined by $Ax^* = b$. At the same time $x = Q\hat{x} + x^*$.

$$f(\hat{x}) = \frac{1}{2}(Q\hat{x} + x^*)^\top A(Q\hat{x} + x^*) - b^\top(Q\hat{x} + x^*)$$



Richardson method is equivalent to the Gradient Descent for quadratics

Consider the following quadratic optimization problem:

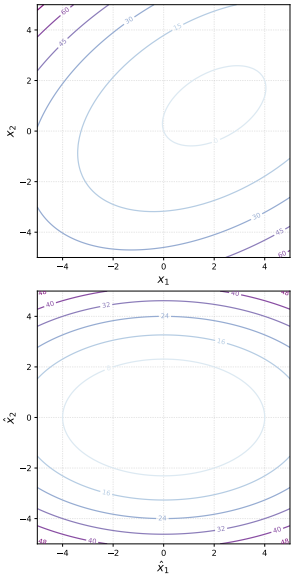
$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}^d_{++}.$$

- Firstly, without loss of generality we can set $c = 0$, which will not affect optimization process.
- Secondly, we have a spectral decomposition of the matrix A:

$$A = Q \Lambda Q^T$$

- Let's show, that we can switch coordinates to make an analysis a little bit easier. Let $\hat{x} = Q^T(x - x^*)$, where x^* is the minimum point of initial function, defined by $Ax^* = b$. At the same time $x = Q\hat{x} + x^*$.

$$\begin{aligned} f(\hat{x}) &= \frac{1}{2}(Q\hat{x} + x^*)^\top A(Q\hat{x} + x^*) - b^\top(Q\hat{x} + x^*) \\ &= \frac{1}{2}\hat{x}^T Q^T A Q \hat{x} + (x^*)^T A Q \hat{x} + \frac{1}{2}(x^*)^T A (x^*)^T - b^T Q \hat{x} - b^T x^* \end{aligned}$$



Richardson method is equivalent to the Gradient Descent for quadratics

Consider the following quadratic optimization problem:

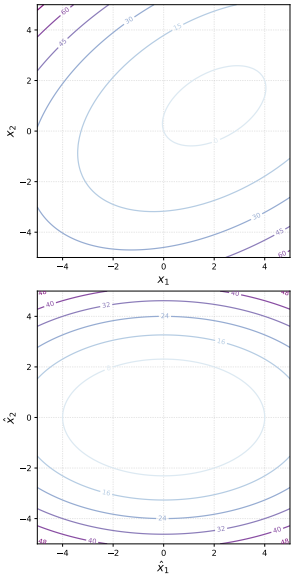
$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top A x - b^\top x + c, \text{ where } A \in \mathbb{S}^d_{++}.$$

- Firstly, without loss of generality we can set $c = 0$, which will not affect optimization process.
- Secondly, we have a spectral decomposition of the matrix A:

$$A = Q \Lambda Q^T$$

- Let's show, that we can switch coordinates to make an analysis a little bit easier. Let $\hat{x} = Q^T(x - x^*)$, where x^* is the minimum point of initial function, defined by $Ax^* = b$. At the same time $x = Q\hat{x} + x^*$.

$$\begin{aligned} f(\hat{x}) &= \frac{1}{2}(Q\hat{x} + x^*)^\top A(Q\hat{x} + x^*) - b^\top(Q\hat{x} + x^*) \\ &= \frac{1}{2}\hat{x}^T Q^T A Q \hat{x} + (x^*)^T A Q \hat{x} + \frac{1}{2}(x^*)^T A (x^*)^T - b^T Q \hat{x} - b^T x^* \\ &= \frac{1}{2}\hat{x}^T \Lambda \hat{x} \end{aligned}$$



Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$x^{k+1} = x^k - \alpha^k \nabla f(x^k)$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$x^{k+1} = x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate}
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$|1 - \alpha\mu| < 1$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
&|1 - \alpha\mu| < 1 \\
&-1 < 1 - \alpha\mu < 1
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T\Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha\lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
&|1 - \alpha\mu| < 1 \\
&-1 < 1 - \alpha\mu < 1 \\
&\alpha < \frac{2}{\mu} \qquad \alpha\mu > 0
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
&|1 - \alpha\mu| < 1 \qquad\qquad |1 - \alpha L| < 1 \\
&-1 < 1 - \alpha\mu < 1 \\
&\alpha < \frac{2}{\mu} \qquad \alpha\mu > 0
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
&|1 - \alpha\mu| < 1 && |1 - \alpha L| < 1 \\
&-1 < 1 - \alpha\mu < 1 && -1 < 1 - \alpha L < 1 \\
&\alpha < \frac{2}{\mu} \qquad \alpha\mu > 0
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 & |1 - \alpha L| &< 1 \\
-1 < 1 - \alpha\mu &< 1 & -1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu &> 0 & \alpha < \frac{2}{L} \qquad \alpha L &> 0
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 & |1 - \alpha L| &< 1 \\
-1 < 1 - \alpha\mu &< 1 & -1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu &> 0 & \alpha < \frac{2}{L} \qquad \alpha L &> 0
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 & |1 - \alpha L| &< 1 \\
-1 < 1 - \alpha\mu &< 1 & -1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu &> 0 & \alpha < \frac{2}{L} \qquad \alpha L &> 0
\end{aligned}
$$

$\alpha < \frac{2}{L}$ is needed for convergence.

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T\Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha\lambda_{(i)}| < 1$$

Remember, that $\lambda_{\min} = \mu > 0, \lambda_{\max} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 \\
-1 < 1 - \alpha\mu &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu &> 0
\end{aligned}
\qquad\qquad
\begin{aligned}
|1 - \alpha L| &< 1 \\
-1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{L} \qquad \alpha L &> 0
\end{aligned}
$$

$\alpha < \frac{2}{L}$ is needed for convergence.

Now we would like to tune α to choose the best (lowest) convergence rate

$$\rho^* = \min_\alpha \rho(\alpha)$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 & |1 - \alpha L| &< 1 \\
-1 < 1 - \alpha\mu &< 1 & -1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu &> 0 & \alpha < \frac{2}{L} \qquad \alpha L &> 0
\end{aligned}
$$

$\alpha < \frac{2}{L}$ is needed for convergence.

Now we would like to tune α to choose the best (lowest) convergence rate

$$\rho^* = \min_\alpha \rho(\alpha) = \min_\alpha \max_i |1 - \alpha \lambda_{(i)}|$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 & |1 - \alpha L| &< 1 \\
-1 < 1 - \alpha\mu &< 1 & -1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu &> 0 & \alpha < \frac{2}{L} \qquad \alpha L &> 0
\end{aligned}
$$

$\alpha < \frac{2}{L}$ is needed for convergence.

Now we would like to tune α to choose the best (lowest) convergence rate

$$
\begin{aligned}
\rho^* &= \min_\alpha \rho(\alpha) = \min_\alpha \max_i |1 - \alpha \lambda_{(i)}| \\
&= \min_\alpha \{|1 - \alpha\mu|, |1 - \alpha L|\}
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha \lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 & |1 - \alpha L| &< 1 \\
-1 < 1 - \alpha\mu &< 1 & -1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu &> 0 & \alpha < \frac{2}{L} \qquad \alpha L &> 0
\end{aligned}
$$

$\alpha < \frac{2}{L}$ is needed for convergence.

Now we would like to tune α to choose the best (lowest) convergence rate

$$
\begin{aligned}
\rho^* &= \min_\alpha \rho(\alpha) = \min_\alpha \max_i |1 - \alpha \lambda_{(i)}| \\
&= \min_\alpha \{|1 - \alpha\mu|, |1 - \alpha L|\} \\
\alpha^* : \quad & 1 - \alpha^* \mu = \alpha^* L - 1
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T\Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha\lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 & |1 - \alpha L| &< 1 \\
-1 < 1 - \alpha\mu &< 1 & -1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu &> 0 & \alpha < \frac{2}{L} \qquad \alpha L &> 0
\end{aligned}
$$

$\alpha < \frac{2}{L}$ is needed for convergence.

Now we would like to tune α to choose the best (lowest) convergence rate

$$
\begin{aligned}
\rho^* &= \min_\alpha \rho(\alpha) = \min_\alpha \max_i |1 - \alpha\lambda_{(i)}| \\
&= \min_\alpha \{|1 - \alpha\mu|, |1 - \alpha L|\} \\
\alpha^* : \quad & 1 - \alpha^*\mu = \alpha^* L - 1 \\
& \alpha^* = \frac{2}{\mu + L}
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T\Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha\lambda_{(i)}| < 1$$

Remember, that $\lambda_{\text{min}} = \mu > 0, \lambda_{\text{max}} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 & |1 - \alpha L| &< 1 \\
-1 < 1 - \alpha\mu &< 1 & -1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu > 0 & & \alpha < \frac{2}{L} \qquad \alpha L > 0 &
\end{aligned}
$$

$\alpha < \frac{2}{L}$ is needed for convergence.

Now we would like to tune α to choose the best (lowest) convergence rate

$$
\begin{aligned}
\rho^* &= \min_\alpha \rho(\alpha) = \min_\alpha \max_i |1 - \alpha\lambda_{(i)}| \\
&= \min_\alpha \{|1 - \alpha\mu|, |1 - \alpha L|\} \\
\alpha^* : \quad & 1 - \alpha^*\mu = \alpha^* L - 1 \\
\alpha^* &= \frac{2}{\mu + L} \qquad \rho^* = \frac{L - \mu}{L + \mu}
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T \Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha\lambda_{(i)}| < 1$$

Remember, that $\lambda_{\min} = \mu > 0, \lambda_{\max} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 & |1 - \alpha L| &< 1 \\
-1 < 1 - \alpha\mu &< 1 & -1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu &> 0 & \alpha < \frac{2}{L} \qquad \alpha L &> 0
\end{aligned}
$$

$\alpha < \frac{2}{L}$ is needed for convergence.

Now we would like to tune α to choose the best (lowest) convergence rate

$$
\begin{aligned}
\rho^* &= \min_\alpha \rho(\alpha) = \min_\alpha \max_i |1 - \alpha\lambda_{(i)}| \\
&= \min_\alpha \{|1 - \alpha\mu|, |1 - \alpha L|\} \\
\alpha^* : \quad & 1 - \alpha^*\mu = \alpha^* L - 1 \\
\alpha^* &= \frac{2}{\mu + L} \qquad \rho^* = \frac{L - \mu}{L + \mu} \\
x^{k+1} &= \left(\frac{L - \mu}{L + \mu}\right)^k x^0
\end{aligned}
$$

Convergence of the ~~Richardson method~~ Gradient Descent

Now we can work with the function $f(x) = \frac{1}{2}x^T\Lambda x$ with $x^* = 0$ without loss of generality (drop the hat from the $\hat{x}$)

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) = x^k - \alpha^k \Lambda x^k \\
&= (I - \alpha^k \Lambda) x^k \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)}) x^k_{(i)} \text{ For } i\text{-th coordinate} \\
x^{k+1}_{(i)} &= (1 - \alpha^k \lambda_{(i)})^k x^0_{(i)}
\end{aligned}
$$

Let's use constant stepsize $\alpha^k = \alpha$. Convergence condition:

$$\rho(\alpha) = \max_i |1 - \alpha\lambda_{(i)}| < 1$$

Remember, that $\lambda_{\min} = \mu > 0, \lambda_{\max} = L \geq \mu$.

$$
\begin{aligned}
|1 - \alpha\mu| &< 1 & |1 - \alpha L| &< 1 \\
-1 < 1 - \alpha\mu &< 1 & -1 < 1 - \alpha L &< 1 \\
\alpha < \frac{2}{\mu} \qquad \alpha\mu &> 0 & \alpha < \frac{2}{L} \qquad \alpha L &> 0
\end{aligned}
$$

$\alpha < \frac{2}{L}$ is needed for convergence.

Now we would like to tune α to choose the best (lowest) convergence rate

$$
\begin{aligned}
\rho^* &= \min_\alpha \rho(\alpha) = \min_\alpha \max_i |1 - \alpha\lambda_{(i)}| \\
&= \min_\alpha \{|1 - \alpha\mu|, |1 - \alpha L|\} \\
\alpha^* : \quad & 1 - \alpha^*\mu = \alpha^* L - 1 \\
\alpha^* &= \frac{2}{\mu + L} \qquad \rho^* = \frac{L - \mu}{L + \mu}
\end{aligned}
$$

$$x^{k+1} = \left(\frac{L - \mu}{L + \mu}\right)^k x^0 \qquad f(x^{k+1}) = \left(\frac{L - \mu}{L + \mu}\right)^{2k} f(x^0)$$

Convergence analysis

So, we have a linear convergence in the domain with rate $\frac{\kappa-1}{\kappa+1} = 1 - \frac{2}{\kappa+1}$, where $\kappa = \frac{L}{\mu}$ is sometimes called *condition number* of the quadratic problem.

κ	ρ	Iterations to decrease domain gap 10 times	Iterations to decrease function gap 10 times
1.1	0.05	1	1
2	0.33	3	2
5	0.67	6	3
10	0.82	12	6
50	0.96	58	29
100	0.98	116	58
500	0.996	576	288
1000	0.998	1152	576

Optimal parameter choice

- The choise of τ that minimizes $\|I - \tau A\|_2$ for $A = A^* > 0$ is (prove it!)

$$\tau_{\text{opt}} = \frac{2}{\lambda_{\min} + \lambda_{\max}}.$$

where $\lambda_{\min}$ is the minimal eigenvalue, and $\lambda_{\max}$ is the maximal eigenvalue of the matrix A.

Optimal parameter choice

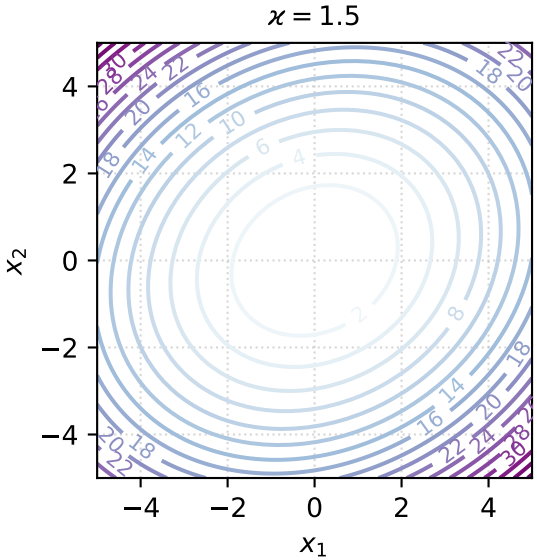
- The choise of τ that minimizes $\|I - \tau A\|_2$ for $A = A^* > 0$ is (prove it!)

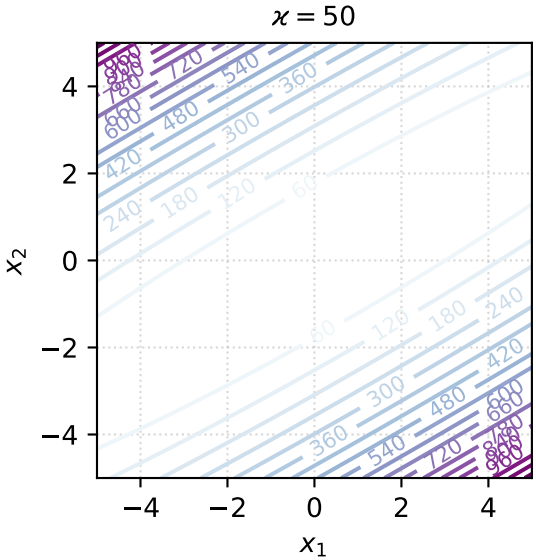
$$\tau_{\text{opt}} = \frac{2}{\lambda_{\min} + \lambda_{\max}}.$$

  where $\lambda_{\min}$ is the minimal eigenvalue, and $\lambda_{\max}$ is the maximal eigenvalue of the matrix A.

- So, to find optimal parameter, we need to know the **bounds of the spectrum** of the matrix A, and we can compute it by using **power method**.

Condition number





Condition number and convergence speed

Even with the optimal parameter choice, the error at the next step satisfies

$$\|e_{k+1}\|_2 \le q\|e_k\|_2, \quad \to \quad \|e_k\|_2 \le q^k\|e_0\|_2,$$

where

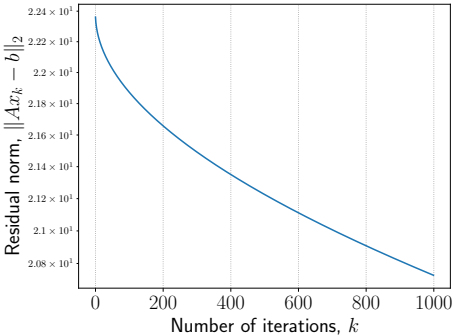
$$q = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}} = \frac{\mathrm{cond}(A) - 1}{\mathrm{cond}(A) + 1},$$

$$\mathrm{cond}(A) = \frac{\lambda_{\max}}{\lambda_{\min}} \quad \text{for} \quad A = A^* > 0$$

is the condition number of A.

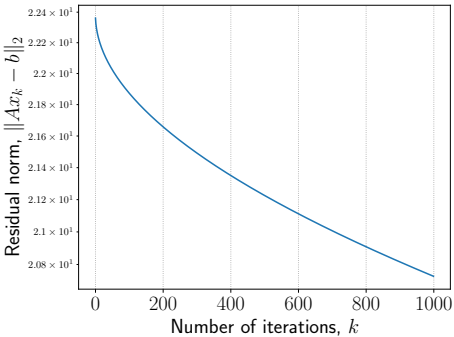
Let us do some demo...

Demo



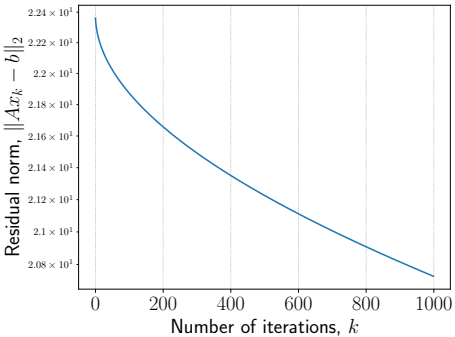
- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.

Demo



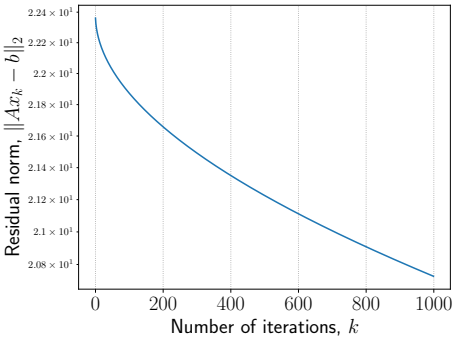
- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.
- This is another reason why **condition number** is so important:

Demo



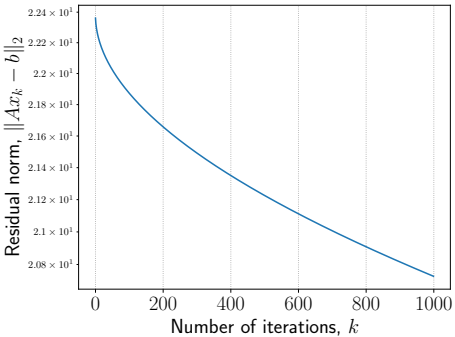
- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.

Demo



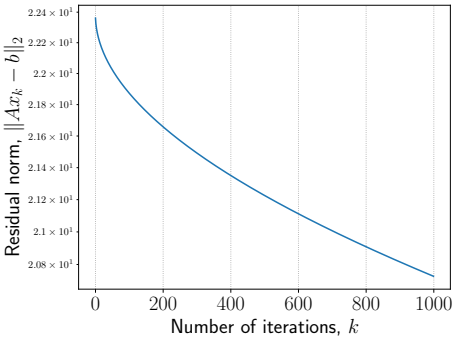
- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.
- Main questions for the iterative method is how to make the matrix **better conditioned**.

Demo



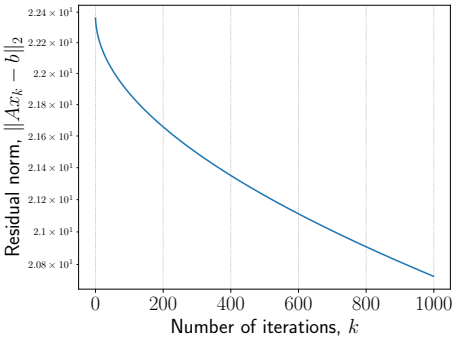
- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.
- Main questions for the iterative method is how to make the matrix **better conditioned**.
- The answer is use preconditioners. Preconditioners will be discussed in further lectures.

Demo



- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.
- Main questions for the iterative method is how to make the matrix **better conditioned**.
- The answer is use preconditioners. Preconditioners will be discussed in further lectures.

Demo



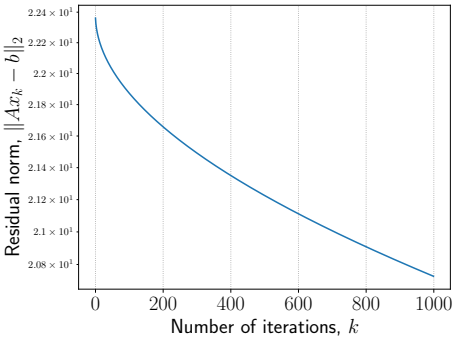
- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.
- Main questions for the iterative method is how to make the matrix **better conditioned**.
- The answer is use preconditioners. Preconditioners will be discussed in further lectures.

Consider non-hermitian matrix A

Possible cases of Richardson iteration behaviour:

- convergence

Demo



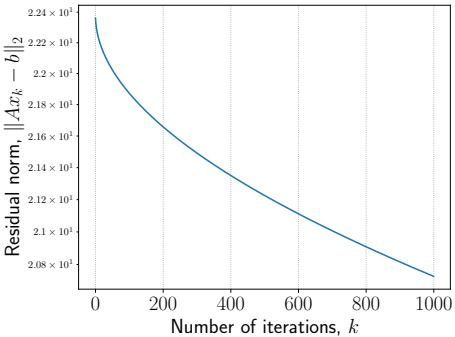
- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.
- Main questions for the iterative method is how to make the matrix **better conditioned**.
- The answer is use preconditioners. Preconditioners will be discussed in further lectures.

Consider non-hermitian matrix A

Possible cases of Richardson iteration behaviour:

- convergence
- divergence

Demo



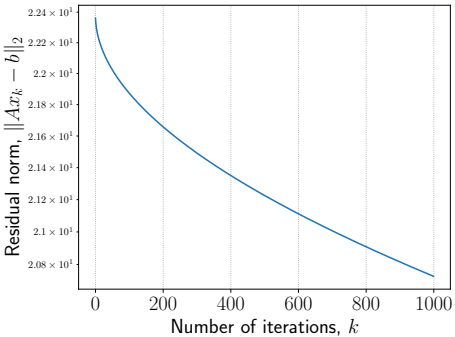
- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.
- Main questions for the iterative method is how to make the matrix **better conditioned**.
- The answer is use preconditioners. Preconditioners will be discussed in further lectures.

Consider non-hermitian matrix A

Possible cases of Richardson iteration behaviour:

- convergence
- divergence
- almost stable trajectory

Demo



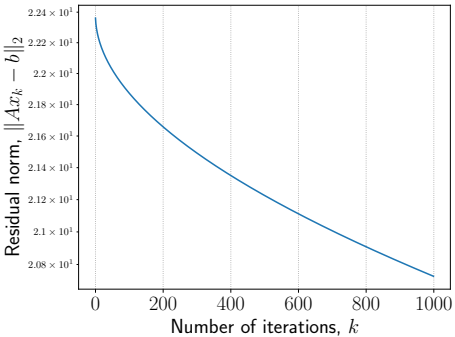
- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.
- Main questions for the iterative method is how to make the matrix **better conditioned**.
- The answer is use preconditioners. Preconditioners will be discussed in further lectures.

Consider non-hermitian matrix A

Possible cases of Richardson iteration behaviour:

- convergence
- divergence
- almost stable trajectory

Demo



- Thus, for **ill-conditioned** matrices the error of the simple iteration method decays very slowly.
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.
- Main questions for the iterative method is how to make the matrix **better conditioned**.
- The answer is use preconditioners. Preconditioners will be discussed in further lectures.

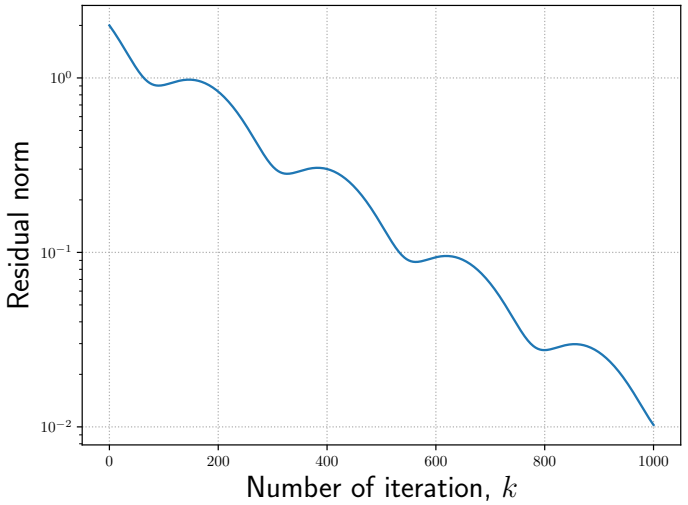
Consider non-hermitian matrix A

Possible cases of Richardson iteration behaviour:

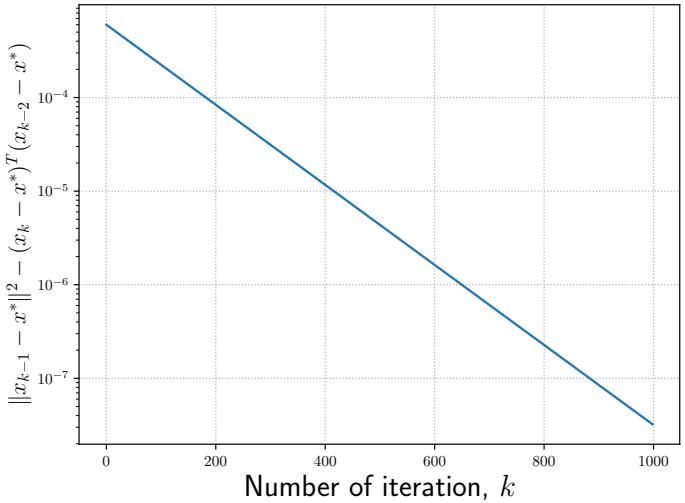
- convergence
- divergence
- almost stable trajectory

Q: how can we identify our case **before** running iterative method?

Spectrum directly affects the convergence



One can still formulate a Lyapunov function [1]



[1] Another approach to build Lyapunov functions for the first order methods in the quadratic case. D. M. Merkulov, I. V. Oseledets

Steepest Descent

Better iterative methods

But before preconditioners, we can use **better iterative methods**.

There is a whole **zoo** of iterative methods, but we need to know just few of them.

Attempt 1: The steepest descent method

- Suppose we **change** τ every step, i.e.

$$x_{k+1} = x_k - \tau_k(Ax_k - f).$$

Attempt 1: The steepest descent method

- Suppose we **change** τ every step, i.e.

$$x_{k+1} = x_k - \tau_k(Ax_k - f).$$

- A possible choice of τ_k is such that it minimizes norm of the current residual

$$\tau_k = \arg\min_{\tau} \|A(x_k - \tau_k(Ax_k - f)) - f\|_2^2.$$

Attempt 1: The steepest descent method

- Suppose we **change** τ every step, i.e.

$$x_{k+1} = x_k - \tau_k(Ax_k - f).$$

- A possible choice of τ_k is such that it minimizes norm of the current residual

$$\tau_k = \arg\min_{\tau} \|A(x_k - \tau_k(Ax_k - f)) - f\|_2^2.$$

- This problem can be solved analytically (derive this solution!)

$$\tau_k = \frac{r_k^\top r_k}{r_k^\top A r_k}, \quad r_k = Ax_k - f$$

Attempt 1: The steepest descent method

- Suppose we **change** τ every step, i.e.

$$x_{k+1} = x_k - \tau_k(Ax_k - f).$$

- A possible choice of τ_k is such that it minimizes norm of the current residual

$$\tau_k = \arg\min_{\tau} \|A(x_k - \tau_k(Ax_k - f)) - f\|_2^2.$$

- This problem can be solved analytically (derive this solution!)

$$\tau_k = \frac{r_k^\top r_k}{r_k^\top A r_k}, \quad r_k = Ax_k - f$$

- This method is called **the steepest descent**.

Attempt 1: The steepest descent method

- Suppose we **change** τ every step, i.e.

$$x_{k+1} = x_k - \tau_k(Ax_k - f).$$

- A possible choice of τ_k is such that it minimizes norm of the current residual

$$\tau_k = \arg\min_{\tau} \|A(x_k - \tau_k(Ax_k - f)) - f\|_2^2.$$

- This problem can be solved analytically (derive this solution!)

$$\tau_k = \frac{r_k^\top r_k}{r_k^\top A r_k}, \quad r_k = Ax_k - f$$

- This method is called **the steepest descent**.
- However, it still converges similarly to the Richardson iteration.

Exact line search aka steepest descent

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}^+} f(x_{k+1}) = \arg\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))$$

More theoretical than practical approach. It also allows you to analyze the convergence, but often exact line search can be difficult if the function calculation takes too long or costs a lot. An interesting theoretical property of this method is that each following iteration is orthogonal to the previous one:

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))$$

Exact line search aka steepest descent

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}^+} f(x_{k+1}) = \arg\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))$$

More theoretical than practical approach. It also allows you to analyze the convergence, but often exact line search can be difficult if the function calculation takes too long or costs a lot. An interesting theoretical property of this method is that each following iteration is orthogonal to the previous one:

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))$$

Optimality conditions:

Exact line search aka steepest descent

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}^+} f(x_{k+1}) = \arg\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))$$

More theoretical than practical approach. It also allows you to analyze the convergence, but often exact line search can be difficult if the function calculation takes too long or costs a lot. An interesting theoretical property of this method is that each following iteration is orthogonal to the previous one:

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))$$

Optimality conditions:

$$\nabla f(x_{k+1})^\top \nabla f(x_k) = 0$$

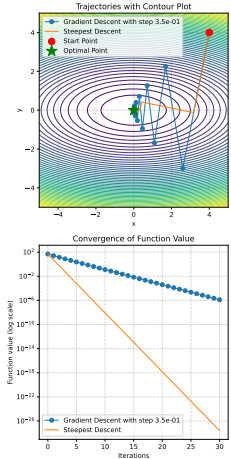


Figure 2: Steepest Descent

Open In Colab ♣

Chebyshev iteration

Attempt 2: Chebyshev iteration

Another way to find τ_k is to consider

$$e_{k+1} = (I - \tau_k A)e_k = (I - \tau_k A)(I - \tau_{k-1}A)e_{k-1} = \ldots = p(A)e_0,$$

where $p(A)$ is a **matrix polynomial** (simplest matrix function)

$$p(A) = (I - \tau_k A)\ldots(I - \tau_0 A),$$

and $p(0) = 1$.

Optimal choice of time steps

The error is written as

$$e_{k+1} = p(A)e_0,$$

and hence

$$\|e_{k+1}\| \leq \|p(A)\|\|e_0\|,$$

where $p(0) = 1$ and $p(A)$ is a **matrix polynomial**.

To get better **error reduction**, we need to minimize

$$\|p(A)\|$$

over all possible polynomials $p(x)$ of degree $k+1$ such that $p(0) = 1$. We will use $\|\cdot\|_2$.

Polynomials least deviating from zeros

Important special case: $A = A^* > 0$.

Then, $A = U\Lambda U^*$,

and

$$\|p(A)\|_2 = \|Up(\Lambda)U^*\|_2 = \|p(\Lambda)\|_2 = \max_i |p(\lambda_i)| \overset{!}{\leq} \max_{\lambda_{\min} \leq \lambda \leq \lambda_{\max}} |p(\lambda)|.$$

The latter inequality is the only approximation. Here we make a **crucial assumption** that we do not want to benefit from the distribution of the spectrum between $\lambda_{\min}$ and $\lambda_{\max}$.

Thus, we need to find a polynomial $p(\lambda)$ such that $p(0) = 1$, and which has the least possible deviation from 0 on $[\lambda_{\min}, \lambda_{\max}]$.

Polynomials least deviating from zeros (2)

We can do the affine transformation of the interval $[\lambda_{\min}, \lambda_{\max}]$ to the interval $[-1, 1]$:

$$\xi = \frac{\lambda_{\max} + \lambda_{\min} - (\lambda_{\min} - \lambda_{\max})x}{2}, \quad x \in [-1, 1].$$

The problem is then reduced to the problem of finding the **polynomial least deviating from zero** on an interval $[-1, 1]$.

Exact solution: Chebyshev polynomials

The exact solution to this problem is given by the famous **Chebyshev polynomials** of the form

$$T_n(x) = \cos(n \arccos x)$$

What do you need to know about Chebyshev polynomials

1. This is a polynomial!

We can plot them...

What do you need to know about Chebyshev polynomials

1. This is a polynomial!
2. We can express T_n from T_{n-1} and T_{n-2}:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad T_0(x) = 1, \quad T_1(x) = x$$

We can plot them...

What do you need to know about Chebyshev polynomials

1. This is a polynomial!
2. We can express T_n from T_{n-1} and T_{n-2}:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad T_0(x) = 1, \quad T_1(x) = x$$

3. $|T_n(x)| \leq 1$ on $x \in [-1, 1]$.

We can plot them...

What do you need to know about Chebyshev polynomials

1. This is a polynomial!
2. We can express T_n from T_{n-1} and T_{n-2}:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad T_0(x) = 1, \quad T_1(x) = x$$

3. $|T_n(x)| \leq 1$ on $x \in [-1, 1]$.
4. It has $(n+1)$ **alternation points**, where the maximal absolute value is achieved (this is the sufficient and necessary condition for the **optimality**) (Chebyshev alternance theorem, no proof here).

We can plot them...

What do you need to know about Chebyshev polynomials

1. This is a polynomial!
2. We can express T_n from T_{n-1} and T_{n-2}:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad T_0(x) = 1, \quad T_1(x) = x$$

3. $|T_n(x)| \leq 1$ on $x \in [-1, 1]$.
4. It has $(n+1)$ **alternation points**, where the maximal absolute value is achieved (this is the sufficient and necessary condition for the **optimality**) (Chebyshev alternance theorem, no proof here).
5. The **roots** are just

$$n \arccos x_k = \frac{\pi}{2} + \pi k, \quad \rightarrow \quad x_k = \cos \frac{\pi(2k+1)}{2n}, \; k = 0, \ldots, n-1$$

We can plot them...

Demo

Convergence of the Chebyshev-accelerated Richardson iteration

Note that $p(x) = (1 - \tau_n x) \dots (1 - \tau_0 x)$, hence roots of $p(x)$ are $1/\tau_i$ and that we additionally need to map back from $[-1, 1]$ to $[\lambda_{\min}, \lambda_{\max}]$. This results into

$$\tau_i = \frac{2}{\lambda_{\max} + \lambda_{\min} - (\lambda_{\max} - \lambda_{\min}) x_i}, \quad x_i = \cos \frac{\pi(2i+1)}{2n} \quad i = 0, \dots, n-1$$

The convergence (we only give the result without the proof) is now given by

$$e_{k+1} \leq C q^k e_0, \quad q = \frac{\sqrt{\mathrm{cond}(A)} - 1}{\sqrt{\mathrm{cond}(A)} + 1},$$

which is better than in the Richardson iteration.

Demo

- Permutation of roots of Chebyshev polynomial has crucial effect on convergence

Demo

- Permutation of roots of Chebyshev polynomial has crucial effect on convergence
- On the optimal permutation you can read in paper (V. Lebedev, S. Finogenov 1971) (ru, en)

Beyond Chebyshev

- We have made an important assumption about the spectrum: it is contained within an interval over the real line (and we need to know the bounds)

Beyond Chebyshev

- We have made an important assumption about the spectrum: it is contained within an interval over the real line (and we need to know the bounds)
- If the spectrum is contained within **two intervals**, and we know the bounds, we can also put the optimization problem for the **optimal polynomial**.

Spectrum of the matrix contained in multiple segments

- For the case of **two segments** the best polynomial is given by **Zolotarev polynomials** (expressed in terms of elliptic functions). Original paper was published in 1877, see details here

Spectrum of the matrix contained in multiple segments

- For the case of **two segments** the best polynomial is given by **Zolotarev polynomials** (expressed in terms of elliptic functions). Original paper was published in 1877, see details here
- For the case of **more than two segments** the best polynomial can be expressed in terms of **hyperelliptic functions**

How can we make it better

- The implementation of the Chebyshev acceleration requires the knowledge of the spectrum.

$$r_k = Ax_k - f.$$

How can we make it better

- The implementation of the Chebyshev acceleration requires the knowledge of the spectrum.
- It only stores the **previous vector** x_k and computes the new correction vector

$$r_k = Ax_k - f.$$

How can we make it better

- The implementation of the Chebyshev acceleration requires the knowledge of the spectrum.
- It only stores the **previous vector** x_k and computes the new correction vector

$$r_k = Ax_k - f.$$

- It belongs to the class of **two-term** iterative methods, i.e. it approximates x_{k+1} using 2 vectors: x_k and r_k.

How can we make it better

- The implementation of the Chebyshev acceleration requires the knowledge of the spectrum.
- It only stores the **previous vector** x_k and computes the new correction vector

$$r_k = Ax_k - f.$$

- It belongs to the class of **two-term** iterative methods, i.e. it approximates x_{k+1} using 2 vectors: x_k and r_k.
- It appears that if we **store more vectors**, then we can go without the spectrum estimation (and better convergence in practice)!

Crucial point: Krylov subspace

The Chebyshev method produces the approximation of the form

$$x_{k+1} = x_0 + p(A)r_0,$$

i.e. it lies in the **Krylov subspace** of the matrix which is defined as

$$\mathcal{K}_k(A, r_0) = \mathrm{Span}(r_0, Ar_0, A^2r_0, \ldots, A^{k-1}r_0)$$

The most natural approach then is to find the vector in this **linear subspace** that minimizes certain **norm of the error**

Idea of Krylov methods

The idea is to minimize given functional: - Energy norm of error for systems with hermitian positive-definite matrices (CG method). - Residual norm for systems with general matrices (MINRES and GMRES methods). - Rayleigh quotient for eigenvalue problems (Lanczos method).

To make methods practical one has to 1. Orthogonalize vectors $A^i r_0$ of the Krylov subspace for stability (Lanczos process). 2. Derive recurrent formulas to decrease complexity.

We will consider these methods in details on the next lecture.

Take home message

- Main idea of iterative methods

Take home message

- Main idea of iterative methods
- Richardson iteration: hermitian and non-hermitian case

Take home message

- Main idea of iterative methods
- Richardson iteration: hermitian and non-hermitian case
- Chebyshev acceleration

Take home message

- Main idea of iterative methods
- Richardson iteration: hermitian and non-hermitian case
- Chebyshev acceleration
- Definition of Krylov subspace

Source

- NLA lecture by prof. Ivan Oseledets