Lecture 4

Eigenvalue problem

(diagonalization of a matrix)

outline

- eigenvalue problem – general remarks
- similarity transformations:
    - symmetric matrices
    - nonsymmetric matrices
- Calculations of eigenvalues and eigenvectors
- General eigenvalue problem
- SVD factorization
- numerical examples

Eigenvalue problem (EVP)

We define (ordinary) EVP as follows

$$n \in Z^+, \quad \vec{x}_k \in C^n, \quad A \in C^{n\times n}, \quad \lambda_k \in C$$

$$\boxed{A\vec{x}_k = \lambda_k \vec{x}_k}$$

our task is to find **the eigenvalue spectrum** of matrix A

$$Sp(A) = \{\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_n\}$$

and corresponding set of **eigenvectors**

$$\{\vec{x}_1, \vec{x}_2, \vec{x}_3, \ldots, \vec{x}_n\}$$

The process of solving EVP is called **diagonalization of matrix**.

Besides ordinary EVP we will consider so called (general) EVP → GEVP

$$A\vec{x}_k = \lambda_k B\vec{x}_k, \qquad B \in C^{n\times n}$$

which mathemathically might be transformed to the ordinary EVP

$$B^{-1}A\vec{x}_k = \lambda_k \vec{x}_k, \qquad \Longrightarrow \qquad C\vec{x}_k = \lambda_k \vec{x}_k$$

Solving both EVP & GEVP are common tasks in physics and in engineering
– simply we need eigenvalues and eigenvectors to solve more complex problems.

Left and right eigenvectors

We must keep in mind that EVP can be defined for the right eigenvectors and the left eigenvectors

$\vec{x}_k$ - right eigenvector

$$A\vec{x}_k = \lambda_k \vec{x}_k$$

$\vec{y}_k$ - left eigenvector

$$\vec{y}_k^H A = \lambda_k \vec{y}_k^H$$

$$A^H \vec{y}_k = \lambda_k^* \vec{y}_k$$

hermitian conjugation

$$\alpha \in C$$

$$(\alpha A)^H = \alpha^* \left(A^T\right)^*$$

$$(\alpha \vec{y})^H = \alpha^* \left(\vec{y}^T\right)^*$$

- left and right eigenvectors are linked their eigenvalue spectra are complex conjugated (if eigenvalues are complex numbers → nonsymmetric & complex symmetric matrices)
- symmetric/hermitian matrix has real eigenvalues and left and right eigenvectors are the identical

$$A^H = A \quad \Longrightarrow \quad \lambda_k^* = \lambda_k \in R \quad \& \quad \vec{y}_k = \vec{x}_k$$

- any left and right eigenvectors are mutually orthogonal if they have different eigenvalues

$$\left.\begin{array}{r} A\vec{x}_k = \lambda_k \vec{x}_k \\ \vec{y}_m^H A = \lambda_m \vec{y}_m^H \end{array}\right\} \quad \Longrightarrow \quad \lambda_m \neq \lambda_k \quad \Longrightarrow \quad \vec{y}_m^H \vec{x} = 0$$

proof

$$\left.\begin{array}{r} (\vec{y}_m^H A)\vec{x}_k = \lambda_m \vec{y}_m^H \vec{x}_k \\ \\ \vec{y}_m^H (A\vec{x}_k) = \lambda_k \vec{y}_m^H \vec{x}_k \end{array}\right\} \quad \Longrightarrow \quad \underbrace{(\lambda_m - \lambda_k)}_{\neq 0} \vec{y}_m^H \vec{x} = 0 \quad \Longrightarrow \quad \vec{y}_m^H \vec{x} = 0$$

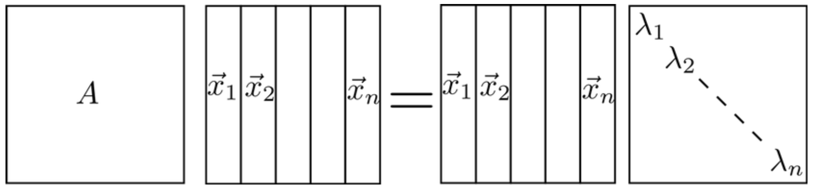
Remark: matrix is an operator which can act on the right or on the left vector

Diagonalization & spectral representation of matrix

We may extend the basic definition so that to involve the whole eigenvalue spectrum and all eigenvectors

$$A\vec{x}_k = \lambda_k \vec{x}_k$$

$$A\left[\vec{x}_1, \vec{x}_2, \vec{x}_3, \ldots, \vec{x}_n\right] = \left[\vec{x}_1, \vec{x}_2, \vec{x}_3, \ldots, \vec{x}_n\right] diag\{\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_n\}$$



From now we use the matrix notation

$$X = \left[\vec{x}_1, \vec{x}_2, \vec{x}_3, \ldots, \vec{x}_n\right]$$
$$\Lambda = diag\{\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_n\}$$

$$\boxed{AX = X\Lambda}$$

Lets assume that the left (Y) and the right vectors (X) are bi-normalized (δ$_{km}$ – Kronecker delta)

$$\boxed{\vec{y}_m^H \vec{x}_k = \delta_{m,k} \quad \Longrightarrow \quad Y^H X = I \quad \Longrightarrow \quad X^{-1} X = I \quad \Longrightarrow \quad X^{-1} = Y^H}$$

then we get: $AX = X\Lambda \qquad Y^H \cdot /$ $\qquad\qquad AX = X\Lambda \qquad / \cdot Y^H$

diagonal form of matrix

$$\boxed{Y^H A X = \Lambda}$$

spectral representation of matrix

$$\boxed{A = X\Lambda Y^H}$$

Spectral representation can be explicitly expressed with set of the eigenvectors

$$A = X\Lambda Y^H = \sum_{k=1}^{n} \lambda_k \vec{x}_k \vec{y}_k^H$$ $\leftarrow$ outer products of two vectors

analogically we can find spectral representation of inverse matrix

$$A^{-1} = \left(X\Lambda Y^H\right)^{-1} = \left(Y^H\right)^{-1}\Lambda^{-1}X^{-1}$$
$$= X\Lambda^{-1}Y^H = \sum_{k=1}^{n} \frac{1}{\lambda_k}\vec{x}_k\vec{y}_k^H$$

Difference between inner (scalar) product and the outer (tensor) product

- inner product gives the scalar (number)

$$\vec{a}^H\vec{b} = c \in C$$

- outer product generates matrix (operator)

$$\vec{a}\vec{b}^H = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} [\beta_1^*, \beta_2^*, \dots, \beta_n^*] = \begin{bmatrix} \alpha_1\beta_1^* & \alpha_1\beta_2^* & \dots & \alpha_1\beta_n^* \\ \alpha_2\beta_1^* & \alpha_2\beta_2^* & \dots & \alpha_2\beta_n^* \\ \dots & \dots & \dots & \dots \\ \alpha_n\beta_1^* & \alpha_n\beta_2^* & \dots & \alpha_n\beta_n^* \end{bmatrix}$$

Projectors – projecting (matrix) operators

The outer product can be used as a part of spectial matrix operator called projector

$$Proj = I - \vec{x}\vec{x}^H$$

When projector act on the vector it projects this vector on the direction established
by the vector used in outer product (below we remove this direction)

$$Proj\,\vec{y} = \left(I - \vec{x}\vec{x}^H\right)\vec{y} = \vec{y} - \vec{x}\underbrace{\left(\vec{x}^H\vec{y}\right)}_{\substack{inner\\product}}$$

Projectors may contain many outer products (many direction), e.g. Gram-Schmidt projector

$$Proj = I - \sum_{k=1}^{n} \vec{x}_k \vec{x}_k^H$$

Similarity transformation of a matrix

There is no method which could allow us directly diagonalize dense square matrix.
Therefore numerical diagonalization becomes a 3-step process

(i) the matrix is transfomed by similarity to a simpler form
(ii) after that diagonalization is easily conducted for the transformed matrix
(iii) then eigenvectors are reconstructed (if required)

Generally the process of diagonalization presents as follows

- matrix transformation → columns are transformed sequentially

$$A = A_0 \rightarrow A_1 \rightarrow A_2 \rightarrow \ldots \rightarrow A_m = B$$

$$A_i = P_i^{-1} A_{i-1} P_i$$ ← i-th similarity transformation step

$$\mathrm{B{=}P}^{-1}AP, \qquad P = P_1 P_2 \ldots P_m$$ ← full similarity matrix

- diagonalization of B with specialized methods

$$B\vec{w}_k = \lambda_k \vec{w}_k, \qquad Sp(A) = Sp(B)$$

- retrieving the eigenvectors of original matrix A

$$\vec{x}_k = P\vec{x}_k$$

Similarity transformation method depends on the type of matrix

- symmetric/hermitian matrix is transformed to tridiagonal matrix

$$\mathrm{A}=\mathrm{A}^H \quad \Longrightarrow \quad T = P^{-1}AP = \begin{bmatrix} * & * & & & & \\ * & * & * & & & \\ & * & * & * & & \\ & & * & * & * & \\ & & & * & * & * \\ & & & & * & * \end{bmatrix}$$

- nonsymmetric matrix is transformed to Hessenberg matrix

$$A \neq A^H \quad \Longrightarrow \quad B = P^{-1}AP = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ & * & * & * & * & * \\ & & * & * & * & * \\ & & & * & * & * \\ & & & & * & * \end{bmatrix}$$

upper Hessenberg matrix

Eigenvalue problem

Example: iterative QR similarity transformation of tridiagonal matrix

$$T := \begin{bmatrix} -2 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -2 \end{bmatrix}$$

- eigenvalues of T

$$\lambda_i := \begin{bmatrix} -0.382 \\ -2.618 \\ -3.618 \\ -1.382 \end{bmatrix}$$

- explicit form of Q_0 & R_0

$$Q := \begin{bmatrix} -0.894 & -0.359 & -0.195 & 0.183 \\ 0.447 & -0.717 & -0.390 & 0.365 \\ 0. & 0.598 & -0.586 & 0.548 \\ 0. & 0. & 0.683 & 0.730 \end{bmatrix} \quad R := \begin{bmatrix} 2.236 & -1.789 & 0.447 & 0. \\ 0. & 1.673 & -1.912 & 0.598 \\ 0. & 0. & 1.464 & -1.952 \\ 0. & 0. & 0. & -0.913 \end{bmatrix}$$

$$T_0 = Q_0 R_0$$
$$T_1 = R_0 Q_0 \quad 1,\ T=, \begin{bmatrix} -2.800 & 0.748 & -1.266\times10^{-16} & 1.454\times10^{-17} \\ 0.748 & -2.343 & 0.875 & 1.828\times10^{-16} \\ 0. & 0.875 & -2.190 & -0.624 \\ 0. & 0. & -0.624 & -0.667 \end{bmatrix}$$

$$T_5 = R_4 Q_4 \quad 5,\ T=, \begin{bmatrix} -3.524 & 0.292 & -9.124\times10^{-17} & -1.352\times10^{-16} \\ 0.292 & -2.701 & 0.118 & 2.505\times10^{-16} \\ 0. & 0.118 & -1.393 & -0.003 \\ 0. & 0. & -0.003 & -0.382 \end{bmatrix}$$

$$T_{10} = R_9 Q_9 \quad 10,\ T=, \begin{bmatrix} -3.614 & 0.063 & -6.479\times10^{-17} & 1.899\times10^{-16} \\ 0.063 & -2.622 & 0.005 & -1.895\times10^{-16} \\ 0. & 0.005 & -1.382 & 4.210\times10^{-6} \\ 0. & 0. & 4.210\times10^{-6} & -0.382 \end{bmatrix}$$

$$T_{15} = R_{14} Q_{14} \quad 15,\ T=, \begin{bmatrix} -3.618 & 0.013 & -6.220\times10^{-17} & -1.993\times10^{-16} \\ 0.013 & -2.618 & 1.903\times10^{-4} & 1.787\times10^{-16} \\ 0. & 1.903\times10^{-4} & -1.382 & -6.790\times10^{-9} \\ 0. & 0. & -6.790\times10^{-9} & -0.382 \end{bmatrix}$$

- each transformation retains tridiagonal form and symmetry within precision of calculations
- eigenvalues start to localize at the right bottom of a matrix
- separated diagonal blocks 2x2 can be diagonalized individually

STEP 1 - Hausholder tridiagonalization of hermitian matrix ($A^H=A$)

$$A = A_0 \to A_1 \to A_2 \to \ldots \to A_m = T$$

$$A_i = P_i^{-1} A_{i-1} P_i$$

$$P_i^{-1} A_{i-1} P_i = P_i A_{i-1} P_i = \left[\begin{array}{c|c|c} J_{i-1} & \vec{c} & 0 \\ \hline \vec{c}^H & \delta_i & \vec{a}_i^H \widetilde{P}_i \\ \hline 0 & \widetilde{P}_i \vec{a}_i & \widetilde{P}_i \widetilde{A}_{i-1} \widetilde{P}_i \end{array}\right] = \left[\begin{array}{cccc|c|c} \delta_1 & \bar{\gamma}_2 & & 0 & 0 & \\ \gamma_2 & \ddots & \ddots & & \vdots & 0 \\ & \ddots & \ddots & \bar{\gamma}_{i-1} & 0 & \\ 0 & & \gamma_{i-1} & \delta_{i-1} & \bar{\gamma}_i & \\ \hline 0 & \ldots & 0 & \gamma_i & \delta_i & \bar{\gamma}_{i+1} 0 \ldots 0 \\ \hline & & & & \gamma_{i+1} & \\ & & & & 0 & \\ & & 0 & & \vdots & \widetilde{P}_i \widetilde{A}_{i-1} \widetilde{P}_i \\ & & & & 0 & \end{array}\right]$$

1

$$P_i = \left[\begin{array}{c|c|c} I & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & \widetilde{P}_i \end{array}\right] \quad \}i-1$$

2

$$\widetilde{P}_i \vec{a}_i = k \cdot \vec{e}_1$$

$$\widetilde{P}_i = I - \beta \vec{u} \vec{u}^H$$

$$\vec{u} = \vec{a} - k \vec{e}_1$$

3

$$\sigma = \|\vec{a}_i\|_2 = \sqrt{\sum_{j=i+1}^{n} |\alpha_{ji}|^2}$$

$$\beta = \begin{cases} \frac{1}{\sigma(\sigma + |\alpha_{i+1,i}|)} & \sigma \neq 0 \\ 0 & \sigma = 0 \end{cases}$$

$$k = -\sigma e^{i\varphi} \iff \alpha_{i+1,i} = e^{i\varphi} |\alpha_{i+1,i}|$$

STEP 2 - iterative similarity transformation with QR factorization (finding eigenvalues)

QR factorization can be easily used to find similarity transformation (below we assume $Q^TQ=I$)

$$T_1 = Q_1 R_1 \qquad Q_1^T \cdot /$$
$$Q_1^T T_1 = R_1 \qquad / \cdot Q_1$$
$$Q_1^T T_1 Q_1 = R_1 Q_1 = T_2$$
$$Q_1^{-1} T_1 Q_1 = T_2$$

- T^{new} – is tridiagonal matrix but with smaller values of off-diagonal elements
- if we perform many such transformation we get matrix "almost" diagonal similar matrix – diagonal elements shall be the eigenvalues

$$T_{k+1} = Q_k^T T_k Q_k = \underbrace{Q_k^T Q_{k-1}^T \ldots Q_1}_{Q^T}\ T\ \underbrace{Q_1^T Q_2^T \ldots Q_k}_{Q} = Q^{-1} T Q$$

- to prevent stagnation of off-diagonal elements the process is accelerated by shifting the matrix with already stabilized eigenvales (these are localizes at the right bottom part of matrix)

$$T_k - \lambda_k I = Q_k R_k$$
$$T_{k+1} = R_k Q_k + \lambda_k I$$

$$\mathrm{T}_n = \left[\begin{array}{ccccc|c} * & * & & & & \\ * & * & * & & & \\ & * & * & * & & \\ & & * & * & * & \\ & & & * & * & 0 \\ \hline & & & & 0 & \lambda_k \end{array}\right]$$

- degenerated eigenvalues form isolated 2x2 blocks which can be diagonalized individually

$$\mathrm{T}_n = \begin{bmatrix} * & & & & & \\ & * & & & & \\ & & * & * & & \\ & & * & * & & \\ & & & & * & \\ & & & & & * \end{bmatrix}$$

But does it make sense?

we know the QR factorization of dense matrix requires $O(n^3)$ operations

QR factorization of tridiagonal matrix

Lets express the QR in the folowing form

$$T = QR \quad \Longrightarrow \quad Q^T T = R \quad \Longrightarrow \quad \underbrace{G_{n-1} \dots G_2 G_1}_{Q_T} T = R$$

- matrices G_k are compounds of Q matrix and each representstransformation of single column in T
- on the right side we have triangular matrix, therefore we need such G_k which eliminate the subdiagonal elements
- G_k are constructed as Givens rotation matrix or Hausholder projector

action of Hauseholder operator: $\boxed{G_k = I - \vec{u}_k \vec{u}_k^H \quad \Longleftrightarrow \quad \vec{u}_k = [0, \dots, c_k, c_{k+1}, 0, \dots, 0]^T}$

$$G_k(G_{k-1} \dots G_1 T) = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & a & b^* & & \\ & & b & -a & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix} \cdot \begin{bmatrix} * & * & * & & & \\ & * & * & & & \\ & & * & * & & \\ & & * & * & * & \\ & & & * & * & * \\ & & & & * & * \end{bmatrix}$$

each requires 2 numbers to remember $\rightarrow$ single QR factorization needs only **2(n-1)** numbers to be saved !!!
(provided that we wish to retrieve the original eigenvectors)

QR factorization of tridiagonal (and Hessenberg) matrices is very efficient and has small small memory requirements

STEP 3 – calcultions of eigenvectors

- from iterative QR factorization we get the matrix B with the eigenvalues on its diagonal (for tridiagonal matrix get the diagonal B while for Hessenberg we get upper triangle matrix B).

- eigenvectors of B are calculated as follows

$$P^{-1}TP = B$$

$$B\vec{y}^{(k)} = \lambda_k \vec{y}^{(k)}$$

$$y_j^{(k)} = \begin{cases} 0 & \Longleftrightarrow \quad j > i \\ 1 & \Longleftrightarrow \quad j = i \\ -\dfrac{\sum\limits_{m=j+1}^{i} b_{j,m} y_m^{(k)}}{b_{j,j} - b_{m,m}} & \Longleftrightarrow \quad j = i-1, i-2, \ldots, 1 \end{cases}$$

$$Y = \left[\vec{y}^{(1)}, \vec{y}^{(2)}, \ldots, \vec{y}^{(n)}\right] = \left[\begin{array}{c|c|c|c|c|c} 1 & * & * & * & * & * \\ & 1 & * & * & * & * \\ & & 1 & * & * & * \\ & & & 1 & * & * \\ & & & & 1 & * \\ & & & & & 1 \end{array}\right]$$

- computation of original vectors $x^{(k)}$ is straightforward, but computationally demanding (the similarity matrix P has to be reconstructed from $Q_1, Q_2, Q_3, \ldots, Q_{n-1}$)

$$P^{-1}AP = B$$
$$A = PBP^{-1}$$

$$\xrightarrow{\quad}$$

$$A\vec{x} = \lambda\vec{x}$$
$$PBP^{-1}x = \lambda x$$
$$B\underbrace{P^{-1}x}_{\vec{y}} = \lambda\underbrace{P^{-1}x}_{\vec{y}}$$
$$\vec{x} = P\vec{y}$$

General (symmetric) eigenvalue problem

As has been mentioned at the beggining of lecture, besides the ordinary EVP we are very often faced to more complex task, namely **general EVP**, which is defined by matrices A and $B \neq I$

$$A\vec{x} = \lambda\, B\vec{x}, \qquad A, B \in R^{n\times n}, \vec{x} \in R^{n\times n}, \lambda \in R$$

GEVP are routinely constructed in Galerkin/Finite Elements Methods when the PDE solution is is approximated with linear combination of smooth basis functions. The entries of B matrix are then the overlap integrals of all pairs of the basis functions, if these are non-orthogonal B becomes non-diagonal matrix.

However, in many cases B, called the mass matrix, is symmetric (hermitian) and positive definite, hence Cholesky factorization ($LL^T \rightarrow LL^H$ for hermitian matrices) can be used in calculations

$$B^H = B \quad \wedge \quad \vec{x}^T B\vec{x} > 0 \qquad \Longrightarrow \qquad B = LL^T$$

We utilize LL^T decomposition to transform GEVP to ordinary EVP.

- first, let's substitute LL^T product into GEVP

$$A\vec{x} = \lambda B\vec{x}$$

$$A\vec{x} = \lambda LL^T\vec{x} \qquad L^{-1}\cdot /$$

$$\underbrace{L^{-1}A\left(L^T\right)^{-1}}_{G}\underbrace{\left(L^T\vec{x}\right)}_{\vec{y}} = \lambda\underbrace{\left(L^T\vec{x}\right)}_{\vec{y}}$$

$$G\vec{y} = \lambda\vec{y}$$

We get ordinary EVP but may now make a question: does such transformation preserve the symmetry of GEVP?

Let's assume both A and B are symmetric and both have their Cholesky factorization

$$A^T = A \quad \wedge \quad B^T = B$$

$$A = UU^T \quad \wedge \quad B = LL^T$$

$$\Longrightarrow$$

$$G = L^{-1} A \left(L^T\right)^{-1}$$

$$G = \underbrace{L^{-1} U}_{S} \underbrace{U^T \left(L^T\right)^{-1}}_{?}$$

$$S = L^{-1} U \quad \Longrightarrow \quad S^T = (L^{-1} U)^T = U^T \left(L^T\right)^{-1}$$

$$G\vec{y} = SS^T \vec{y} = \lambda \vec{y}$$

Transformation keeps the symmetry of original problem. How to find G?

$$G = L^{-1} \underbrace{A \left(L^T\right)^{-1}}_{F} \quad \Longrightarrow \quad LG = F$$

← then use F as the rhs columns
to find columns of G

$$F = A \left(L^T\right)^{-1} \quad \Longrightarrow \quad FL^T = A \quad \Longrightarrow \quad LF^T = A$$

← first find matrix F

In order to find F or G there must be solved n systems of linear equations.
Here is revealed the great advantage of using the matrix decompositon,
only one factorization is needed for the n right hand sides.

After solving EVP with G matrix we obtain the eigenvectors for the original GEVP
by transforming the G eigenvectors

$$\vec{y} = L^T \vec{x}$$

Example: solution of time-independent Schrodinger equation with Finite Difference Method

- **theoretical model**

energy operator:

$$\widehat{H} = -\frac{1}{2m^*}\frac{d^2}{dx^2} + V(x)$$

$$V(x) = \begin{cases} 0 & \Longleftrightarrow \quad x \in [0, x_{max}] \\ +\infty & \Longleftrightarrow \quad x \ni [0, x_{max}] \end{cases}$$

eigenproblem of differential operator:

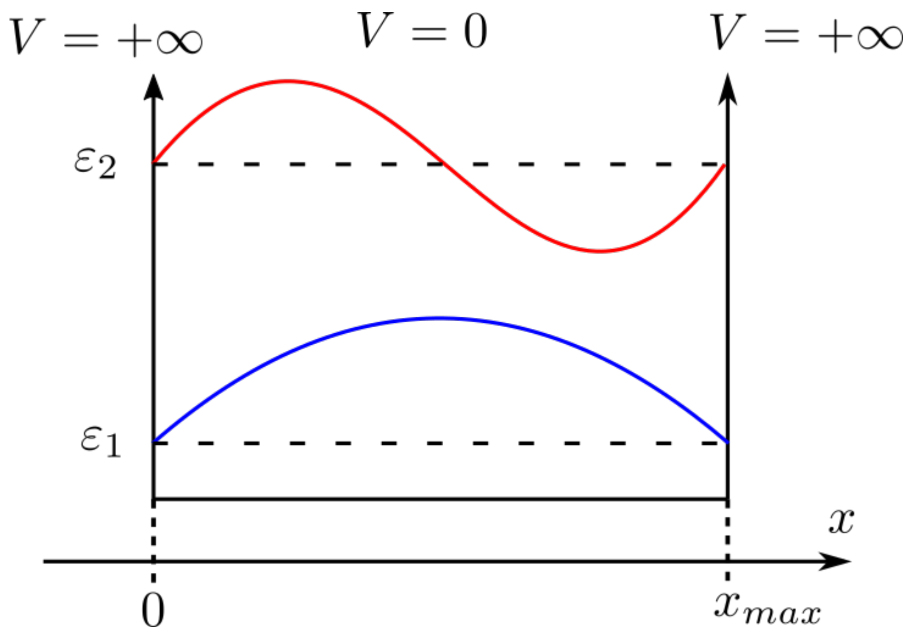
$$\widehat{H}(x)\psi_k(x) = \varepsilon_k \psi_k(x)$$

exact solution - eigenvector + eigenvalue:

$$\psi_k(x) = \sqrt{\frac{2}{x_{max}}}\sin\left(\frac{n\pi x}{x_{max}}\right)$$

$$\varepsilon_n = n^2\frac{\pi^2\hbar^2}{2x_{max}^2 m^*}$$

Sketch of confining potential with two lowest energy solutions

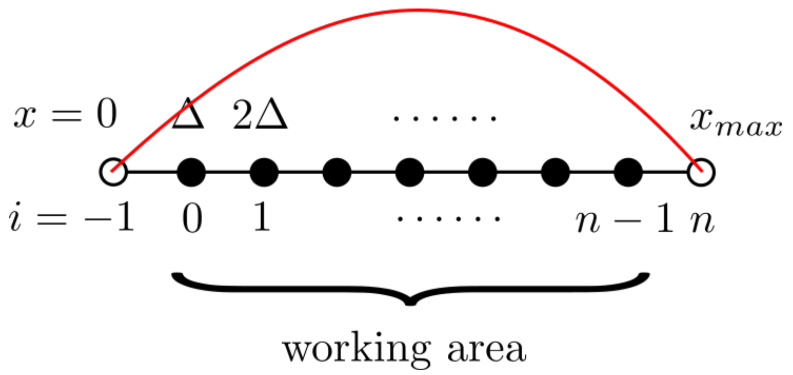


Eigenvalue problem

- **numerical model**

  - mesh of nodes discretizes the space

$$x \to x_i = (i+1)\Delta, \quad i = -1, \underbrace{0, 1, \ldots, n-1}_{\substack{working \\ area}}, n$$



  - distance between neighbouring nodes

$$\Delta = \frac{x_{max}}{n+1}$$

  - we are looking for functions' values at nodes

$$V(x) \to V(x_i) = V_i$$

$$\psi(x) \to \psi(x_i) = \psi_i$$

- finite difference replaces the 2-nd order derivative

$$\frac{d^2\psi}{dx^2} \approx \frac{\psi_{i+1} - 2\psi_i + \psi_{i-1}}{\Delta^2} \quad (+O(\Delta^2))$$

Eigenvalue problem

- discretized energy operator acting at i-th node on the wave function

$$\widehat{H}_d\psi_i = -\frac{1}{2m^*}\frac{\psi_{i+1} - 2\psi_i + \psi_{i-1}}{\Delta^2} + V_i\psi_i = \varepsilon\psi_i$$

$$h_{i,i} = \frac{1}{m^*\Delta^2} + V_i$$ ← diagonal elements

$$h_{i,i-1}\psi_{i-1} + h_{i,i}\psi_i + h_{i,i+1}\psi_{i+1} = \varepsilon\psi_i$$

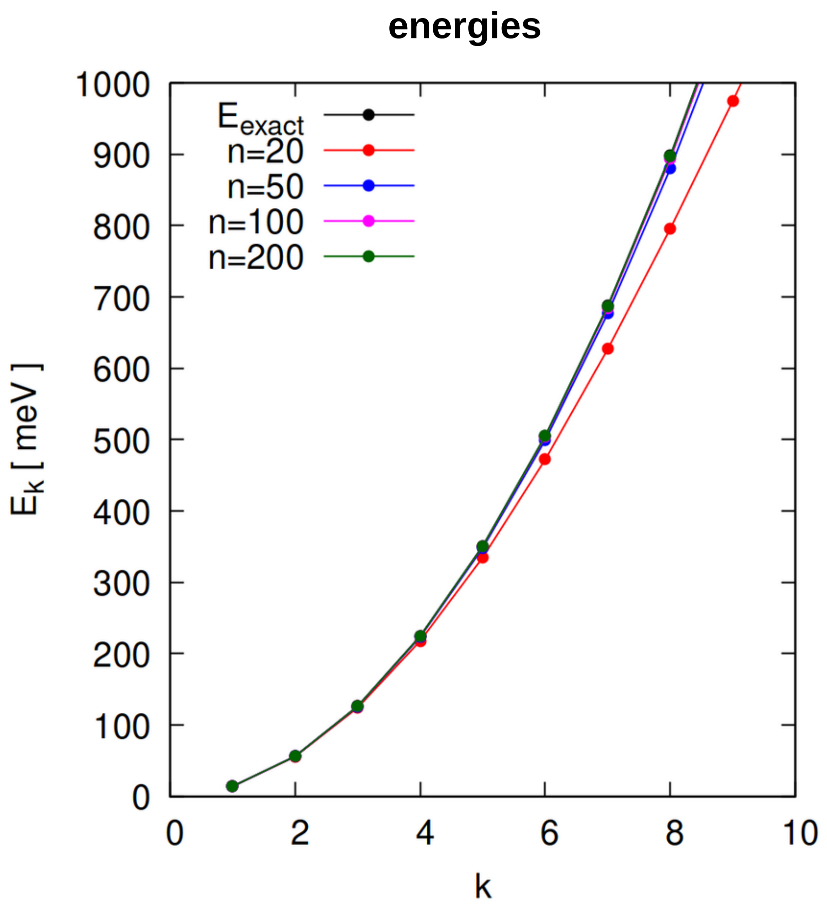
$$h_{i-1,i} = h_{i,i+1} = \frac{-1}{2m^*\Delta^2}$$ ← off-diagonal elements

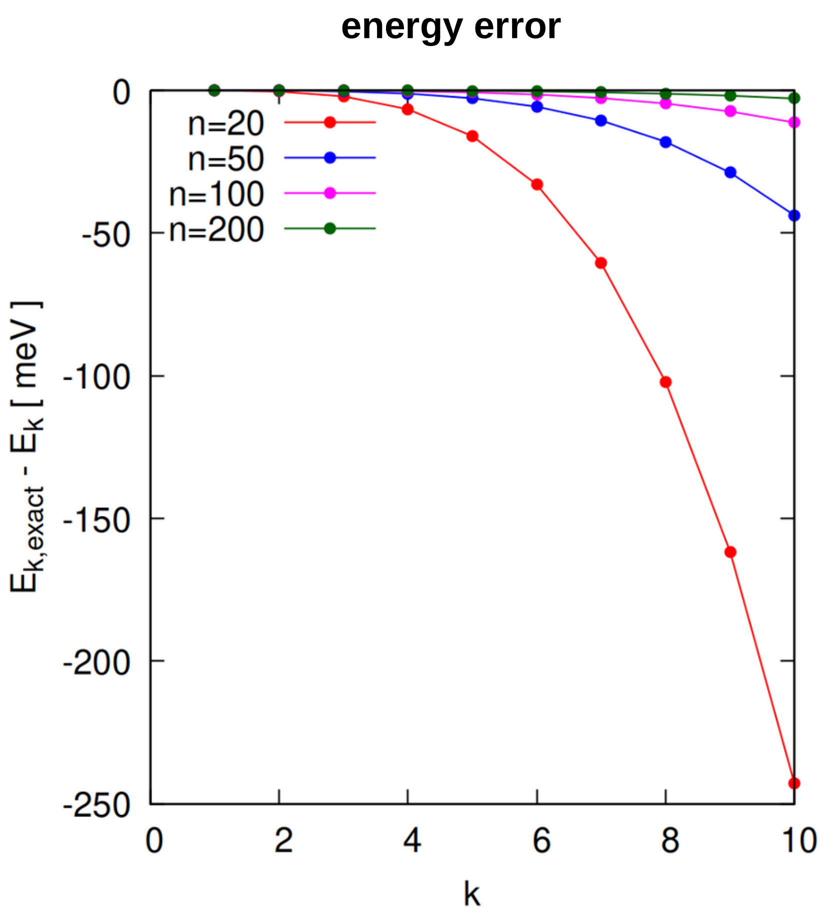
- after writing this expresson for every node in working area we get matrix EVP

$$\begin{bmatrix} h_{11} & h_{1,2} & & & & \\ h_{21} & h_{2,2} & h_{2,3} & & & \\ & h_{32} & h_{3,3} & h_{3,4} & & \\ & & \ddots & \ddots & \ddots & \\ & & & h_{n-2,n-3} & h_{n-2,n-2} & h_{n-2,n-1} \\ & & & & h_{n-1,n-2} & h_{n-1,n-1} \end{bmatrix} \cdot \begin{bmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \\ \vdots \\ \psi_{n-2} \\ \psi_{n-1} \end{bmatrix} = \varepsilon \cdot \begin{bmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \\ \vdots \\ \psi_{n-2} \\ \psi_{n-1} \end{bmatrix}$$

- matrix is **tridiagonal real-symmetric** → the eigenvalues are real
- diagonalization is performed with LAPACK procedure *dstev*

lapack_int LAPACKE_dstev (int matrix_layout, char jobz, lapack_int n,
double* d, double* e, double* z, lapack_int ldz);

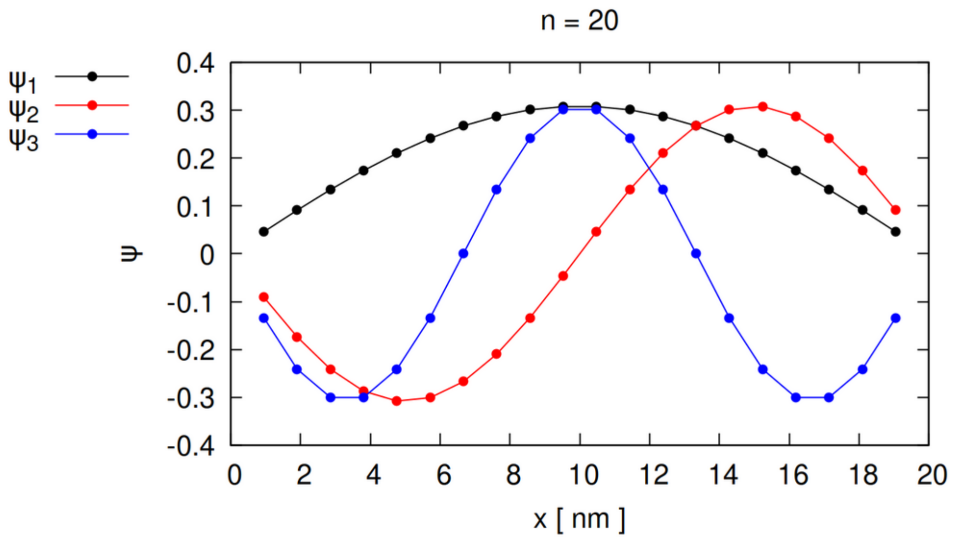


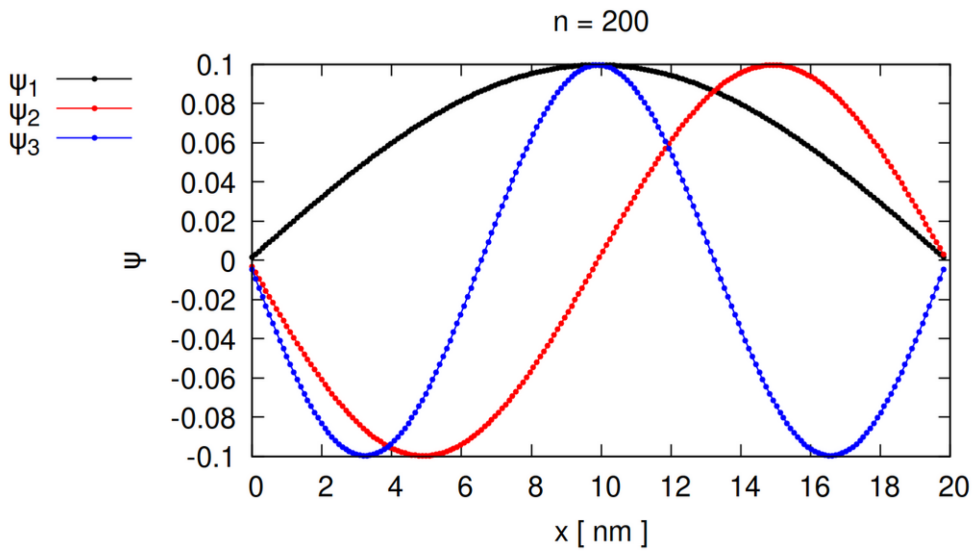


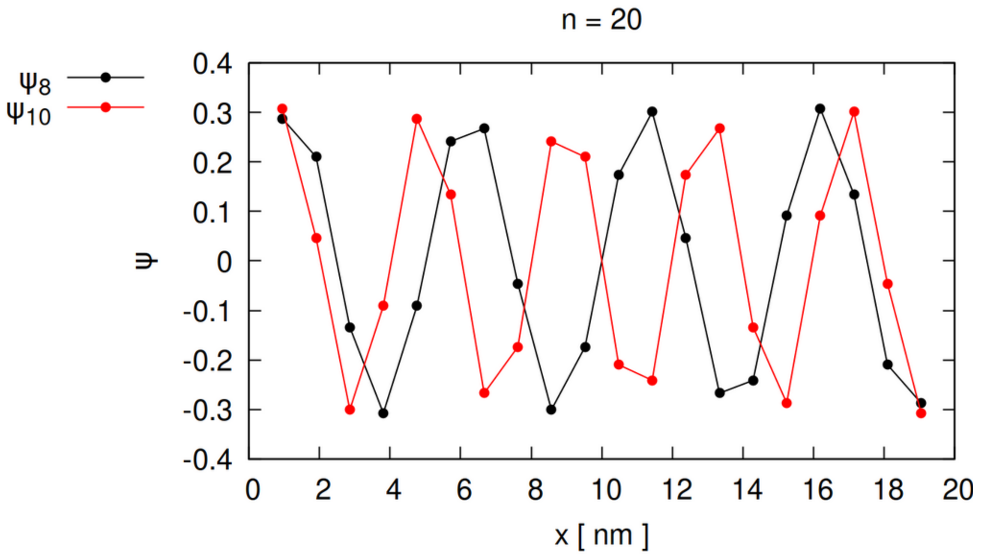
exact eigenvalues

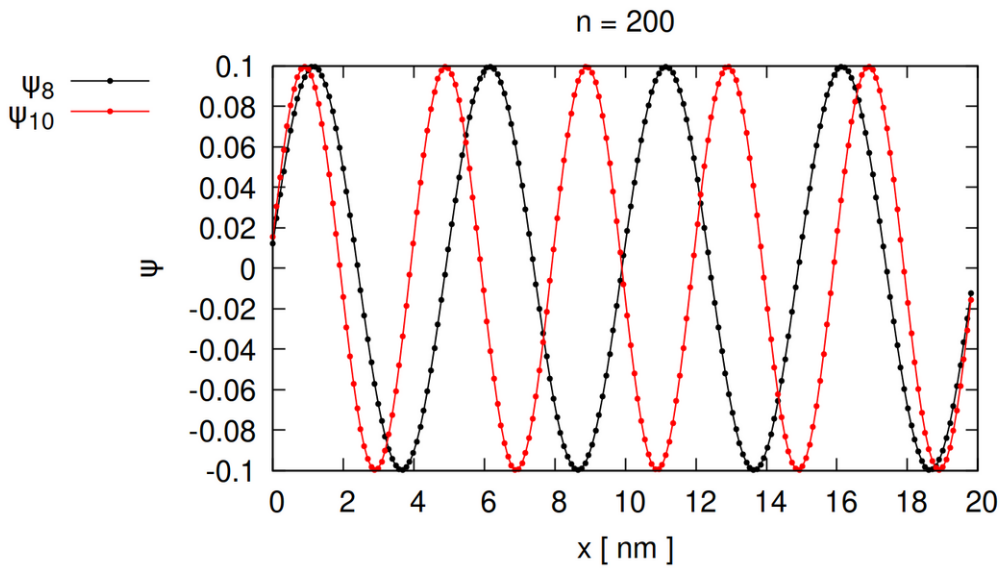
$$\varepsilon_k = k^2 \frac{\pi^2 \hbar^2}{2 x_{max}^2 m^*}$$

wave functions



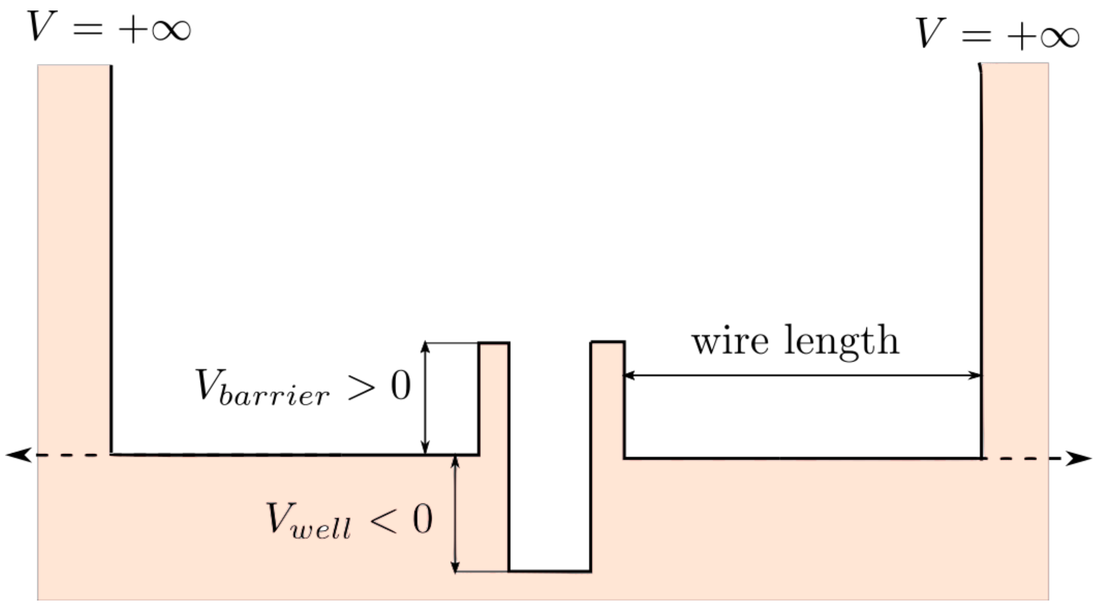






Example: quantum well in quantum well – searching long-lifetime resonances

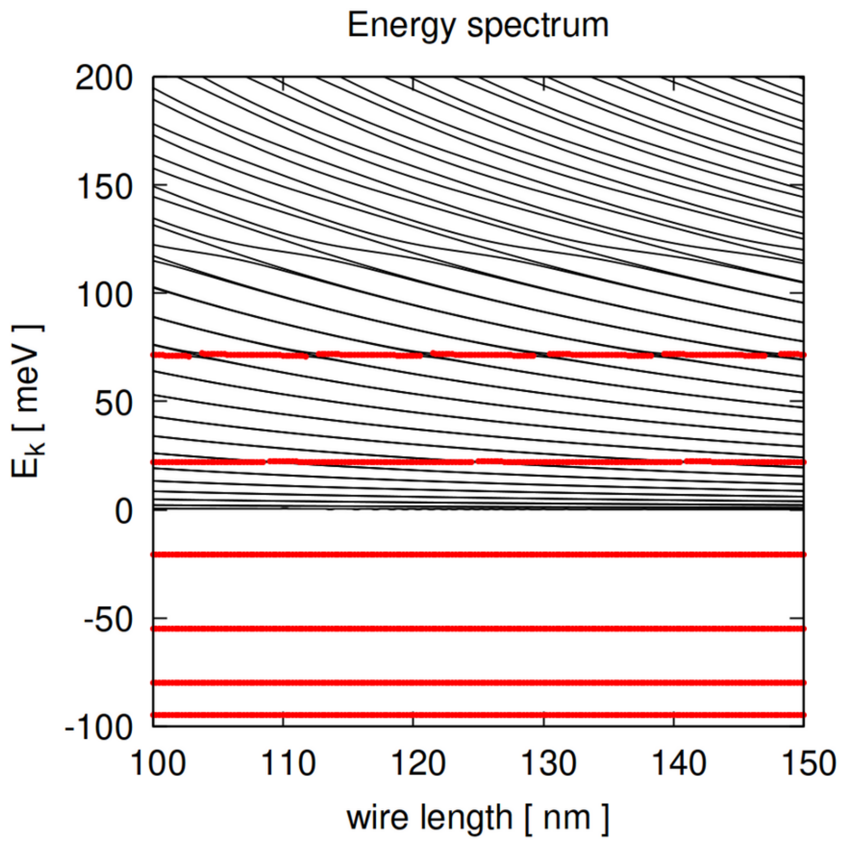
- we may extend the model of single QW by adding the second well surrounding by left and right barriers at the center of system
- this will change only the potential V(x) and consequently only the diagonal elements of matrix

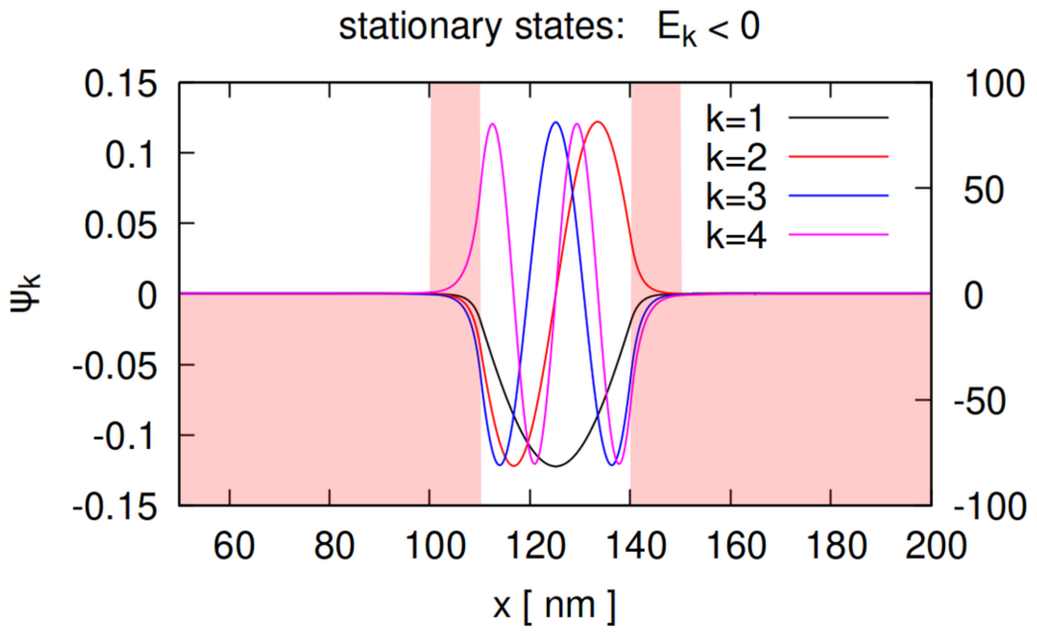


Diagonalization was performed for parameters:

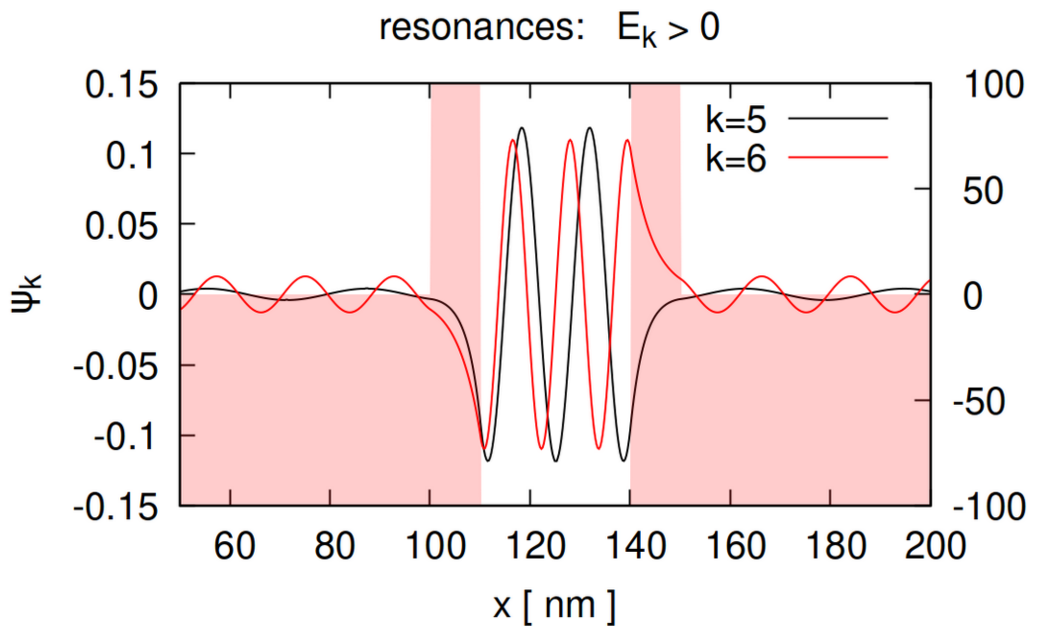
$$\begin{aligned} m^* &= 0.067\, m_0 \qquad (m_0 \text{ - mass of bare electron}) \\ V_{barrier} &= 100 \text{ meV} \\ V_{well} &= -100 \text{ meV} \\ \text{well width} &= 30 \text{ nm} \\ \text{barrier thickness} &= 10 \text{ nm} \\ \text{wire length} &= 100 - 200 \text{ nm} \end{aligned}$$

Expanding box - cont'd





$$E_5 = 22.1meV \quad E_6 = 71.4meV$$



Example: quantum well in quantum well + Complex Absorbing Potential

We may add imaginary smoothly decreasing potential to Hamiltonian at the left and the right boundaries. Characteristic feature of complex potential:

- positive imaginary value creates particle
- negative imaginary value absorbs particle (present case)

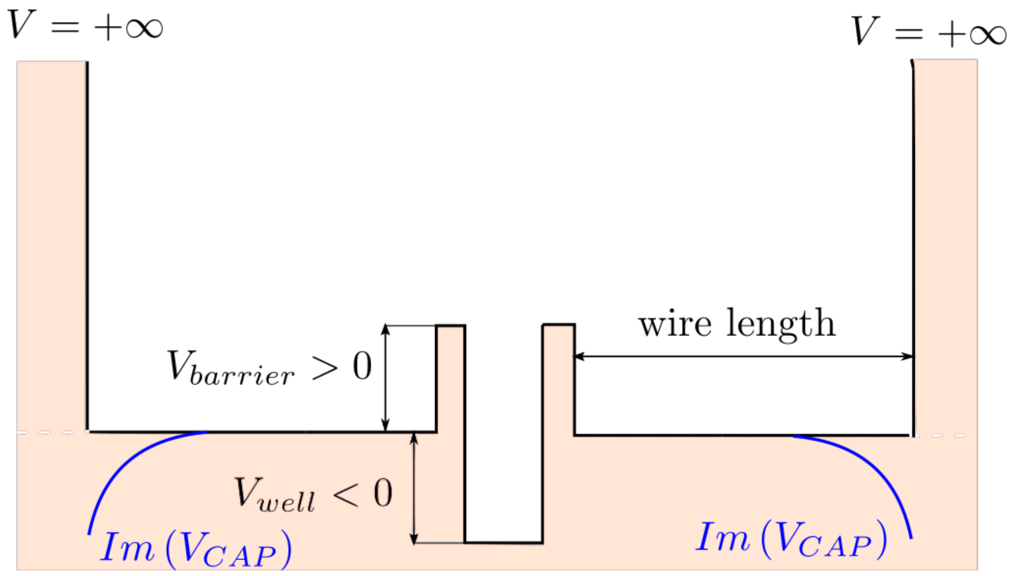
$\rightarrow$ wave function vanishes at boundaries as if particle escapes the box, imitating the open boundary condition i.e. infinite quantum wire

- only potential is modified (diagonal elements of matrix)

$$V_{tot} = V(x) + V_{CAP}(x)$$

$$V_{CAP}(x) = -iV_{max}\left|\frac{x - x_{CAP}}{\Delta_{CAP}}\right|^q, \quad q = 4, 6$$

Δ_{CAP} - range of absorbing potential (blue curve in figure)

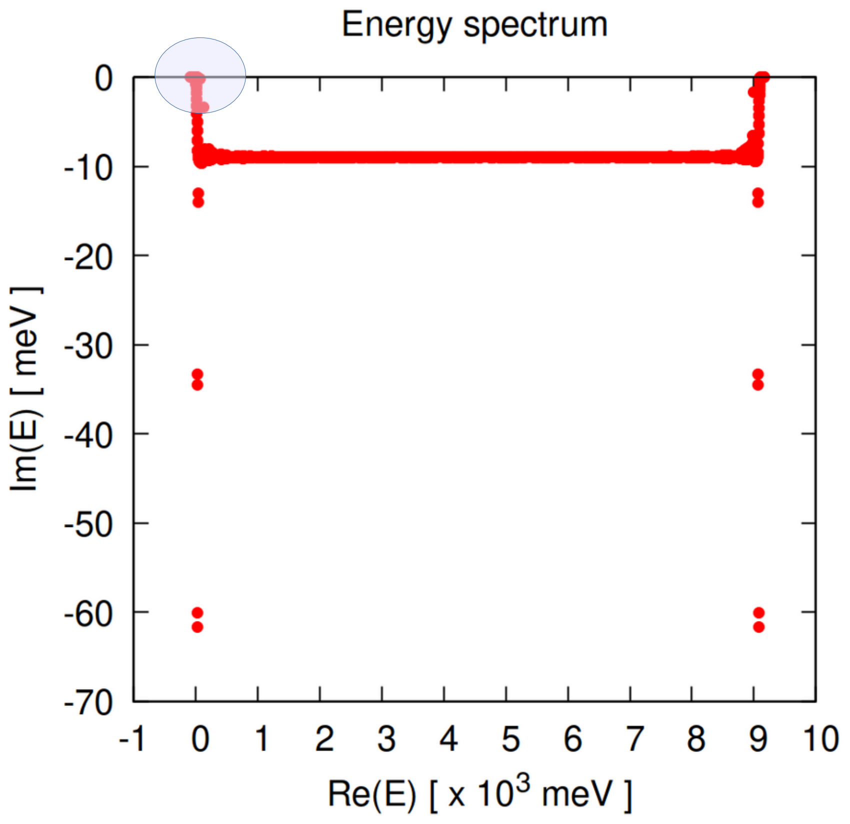


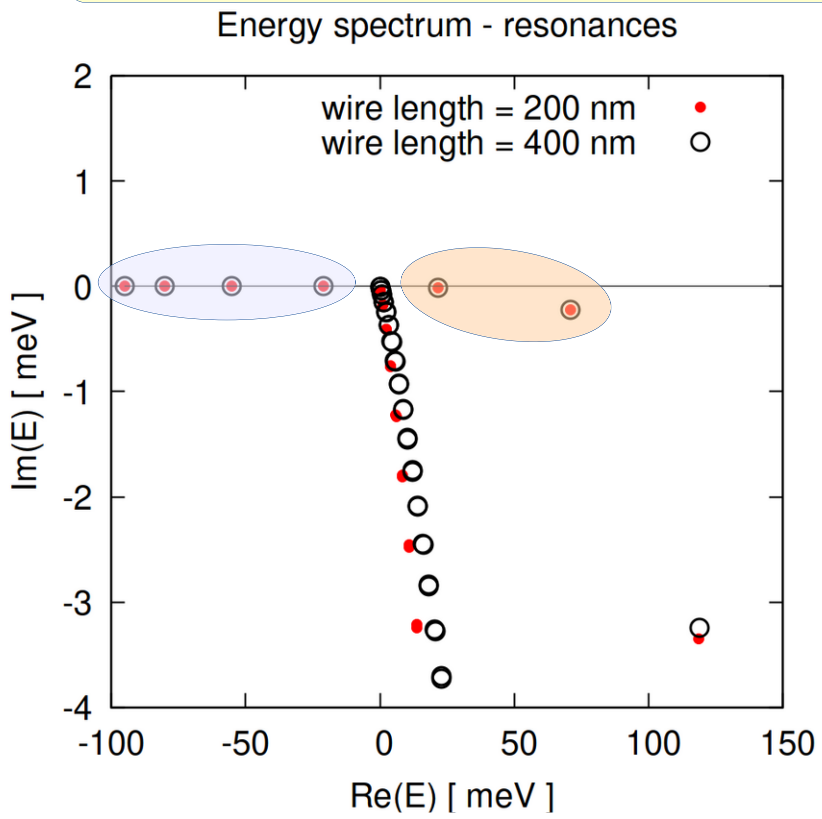
Because we put the complex numbers on diagonal of matrix it becomes nonsymmetric, we must change diagonalization procedure $\rightarrow$ left and right eigenvectors will be different & eigenvalues becomes the complex numbers

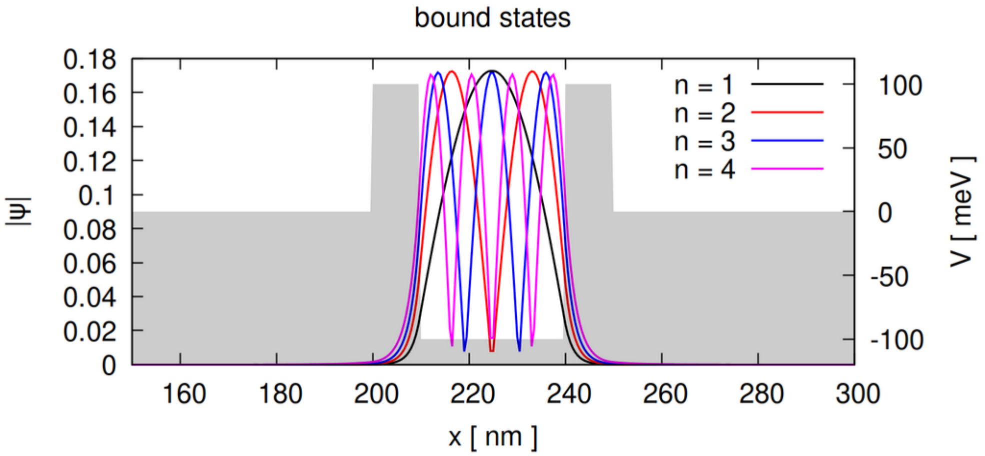
call zgeev(jobvl, jobvr, n, a, lda, w, vl, ldvl, vr, ldvr, work, lwork, rwork, info)

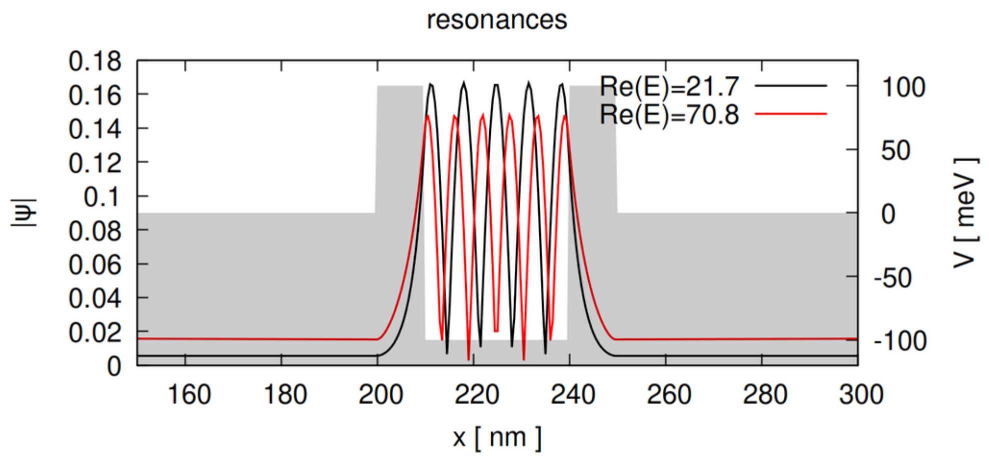
quantum well + absorbing potential → cont'd

bound states and resonances are resistant to changes in quantum well/wire width









$|E_I| < 10^{-12} meV$ - numerical errors

$|E_I| > 10^{-3} meV$ - resonances

Singular Value Decomposition (SVD)

Any rectangular matrix can be decomposed into product of three matrices

$$A = U\Sigma V^H, \qquad (^H\text{- transpose + complex conjugation})$$

$$A \in C^{m\times n}, \quad r = rank(A)$$

$$U \in C^{m\times m}, \quad \Sigma \in R^{m\times n}, \quad V \in C^{n\times n}$$

- matrices U and V are **orthonormal**

$$U^H U = I \qquad V^H V = I$$

- matrix Σ is real with non-negative entries (**singular values**) on its diagonal

$$U^H A V = \Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_p) \in R^{m\times n}, \qquad \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$$

$\mathbf{n > m}$

$$\text{A}= \begin{pmatrix} u_{11} & u_{12} & \dots & u_{1m} \\ u_{21} & u_{22} & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ u_{m1} & \dots & \dots & u_{mm} \end{pmatrix} \times \left(\begin{array}{cccc|cc} \sigma_1 & 0 & \dots & 0 & 0 & 0 \\ 0 & \sigma_2 & \dots & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 & 0 \\ 0 & 0 & \dots & \sigma_n & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & & \\ 0 & 0 & 0 & 0 & & \end{array}\right) \times \begin{pmatrix} v_{11} & v_{12} & \dots & v_{1n} \\ v_{21} & v_{22} & \dots & v_{2n} \\ \vdots & & \ddots & \vdots \\ v_{n1} & v_{n2} & \dots & v_{nn} \end{pmatrix}^H$$

$\mathbf{m > n}$

Eigenvalue problem

How to find SVD?

$$U\Sigma V^H = A \qquad / \cdot V \qquad\qquad\qquad U\Sigma V^H = A \qquad U^H \cdot /$$

$$U\Sigma = AV \quad A^H \cdot / \qquad\qquad\qquad \Sigma V^H = U^H A \qquad /^H$$

$$\begin{tikzcd}
A^H U\Sigma = A^H AV \arrow[dr] & & V\Sigma^T = A^H U \arrow[dl] \\
& \boxed{A^H AV = V\Sigma^T\Sigma} &
\end{tikzcd}$$

we get the standard eigenvalue problem for matrix A^{H}A

$$\boxed{A^H A\vec{v}_j = \sigma_j^2 \vec{v}_j}$$

← for dense matrices bidiagonalization methods are used while for the sparse ones an iterative Lanczos method is employed (A^{H}A – hermitian)

← only for largest singular values, the smallest ones might vanish after squaring

Having V and Σ finding the first p column (thin SVD) is straightforward

$$A = U\Sigma V^T \qquad / \cdot V$$

$$AV = U\Sigma \qquad / \cdot \Sigma^{-1} \quad (\text{ for } \sigma_i > 0)$$

$$\boxed{U = AV\Sigma^{-1}} \quad \Longrightarrow \quad \boxed{\vec{u}_j = \frac{1}{\sigma_j} A\vec{v}_j, \quad j = 1, 2, \ldots, p}$$

Thin SVD – from the two vector subspaces spanned by the columns of U and the columns of V we need only p = rank(A) linearly independent columns to reconstruct the matrix A (in other words – not all U/V columns are always needed)

$$U \in C^{m\times m} \to U_1^{m\times p}$$
$$V \in C^{n\times n} \to V_1^{n\times p}$$
$$\Sigma \in R^{m\times n} \to \Sigma_1 \in R^{p\times p}$$

$$\boxed{A = U_1 \Sigma_1 V_1^H = \sum_{i=1}^{p} \sigma_i \vec{u}_i \vec{v}_i^H}$$

← rank(p) matrix A

A is a rank(p) matrix built as a sum of rank(1) matrices (outer product of two vectors).
Each rank(1) matrix is scaled with singular value, hence the largest contributions gives
the pairs of vectors (u,v) belonging to largest sigmas.
By limiting the summation just to largest k contribution we get rank(k) matrix which approximates matrix A

$$A \approx A^{(k)} = \sum_{i=1}^{k<p} \sigma_i \vec{u}_i \vec{v}_i^H$$

← rank(k) matrix $A^{(k)}$

Applications of SVD

- solving systems of linear equations – standard and overdetermined, backsubstitiution is not required,
  for square matrices it is less computationally efficient than other methods

standard SLE:

$$\begin{aligned} A\vec{x} &= \vec{b}, \\ U\Sigma V^T\vec{x} &= \vec{b}, \qquad U^T \cdot / \\ (U^TU)\Sigma V^T\vec{x} &= U^T\vec{b}, \qquad \Sigma^{-1} \cdot / \\ V^T\vec{x} &= \Sigma^{-1}U^T\vec{b}, \qquad V \cdot / \\ \vec{x} &= V\Sigma^{-1}U^T\vec{b}, \end{aligned}$$

overdetermined SLE:

$$\begin{aligned} A\vec{x} &= \vec{b}, \qquad A^T \cdot / \\ A^TA\vec{x} &= A^T\vec{b}, \quad (A^TA)^{-1} \cdot / \\ \vec{x} &= A^I\vec{b}, \end{aligned}$$

pseudoinverse of (rectangular) matrix

$$A^I = (A^TA)^{-1}A^T$$

$$m > n: \quad A = U\begin{pmatrix}\Sigma \\ 0\end{pmatrix}V^T \quad \Longrightarrow \quad A^I = V\left(\Sigma^{-1} 0\right)U^T \quad \Longrightarrow \quad \vec{x} = V\left(\Sigma^{-1} 0\right)U^T\vec{b}$$

some advanced applications

- statistics: Principal Component Analysis
- speech recognition
- information transmission in wireless communication
- tensor decomposition $\rightarrow$ to speed up work of neural networks
- recommendations systems (trading/commerce)
- classifications of handwritten digits for Optical Character Recognition (OCR)
- facial recognition
- in quantum physcis to assess the degree of entanglement

A survey of SVD applications contains following paper

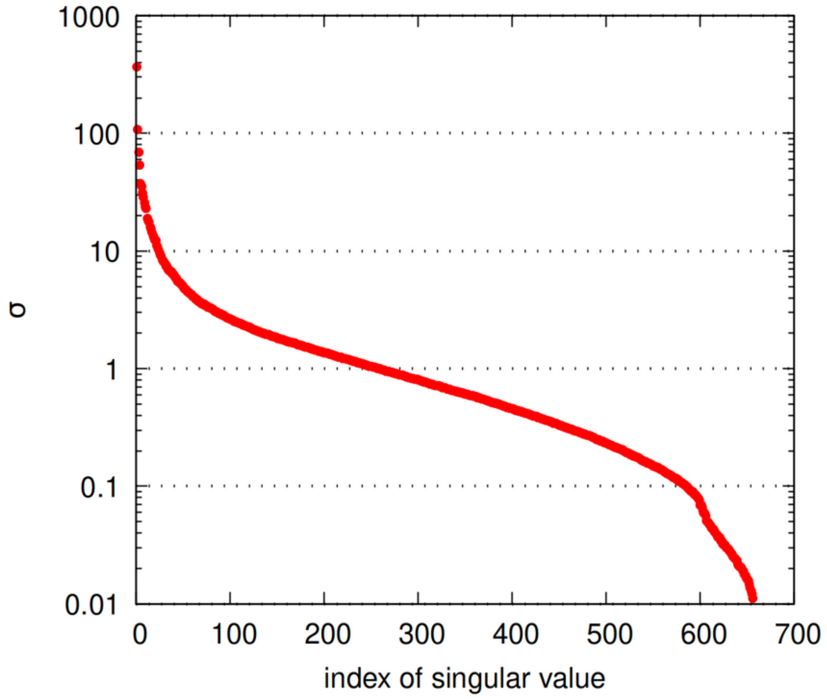
Carla D. Martin and Mason A. Porter, „The Extraordinary SVD”,
https://arxiv.org/pdf/1103.2338

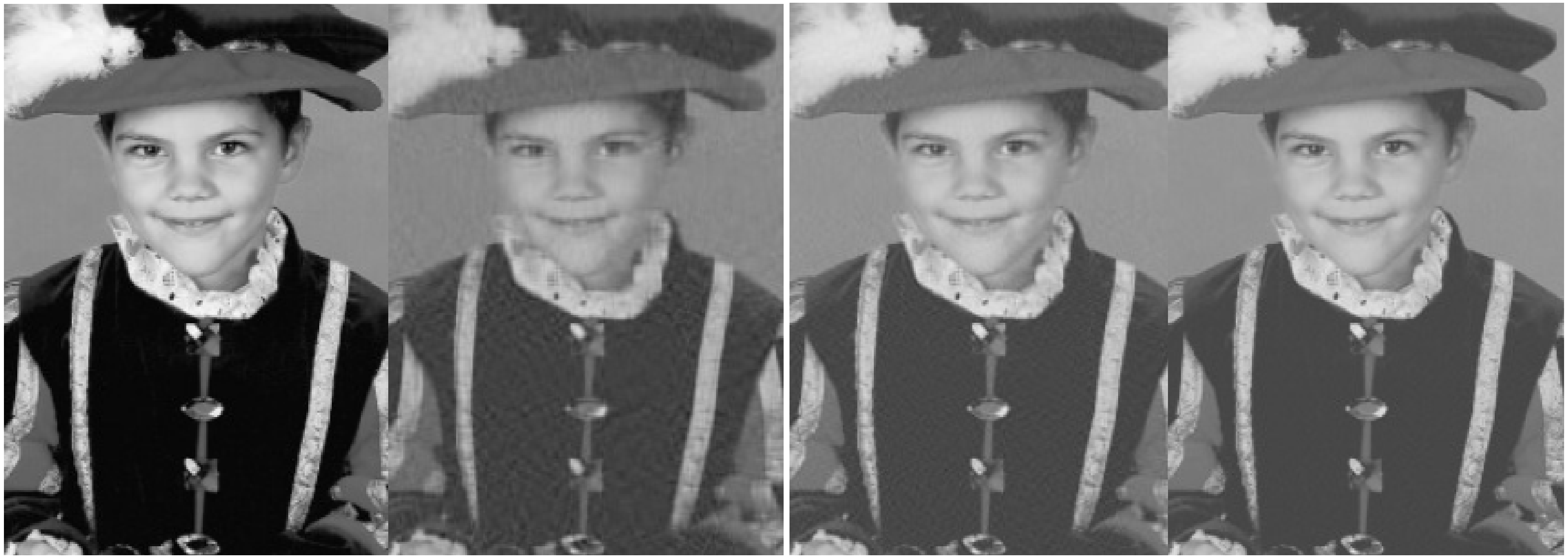
- **compression of images (data gathered in matrix form)**

We may think of an image as a matrix, and each matrix element corresponds to distinct pixel. Full matrix represents an original image, but if we approximate it with rank(k) matrix by retainig small number of vector pairs (u,v) we can save then smaller amount of data at the cost of little (?) worse quality of reconstructed picture.

$$A^{(k)} = \sum_{j=1}^{k<p} \sigma_j u_j v_j^T$$

$$A \in R^{m\times n}, \quad m = 890, \quad n = 650$$





full rank

$A^{(50)}$

$A^{(100)}$

$A^{(200)}$