Queen Mary
University of London

HIGH-ACCURACY NUMERICAL METHODS IN GENERAL RELATIVITY

Rodrigo Panosso Macedo

(Part II)

r.panossomacedo@qmul.ac.uk

The 4th Afro-Franco-Brazilian meeting on Mathematics and Physics - July 2017

SUMMARISE I

- **Spectral representation:** high accuracy approximation of an analytical function

$$f(x) \approx \sum_{i=0}^{N} c_i^{(N)} \phi_i(x)$$

- **Basis Function:** trigonometric if function is periodic (Fourier), Chebyshev else

- **Collocation Method:** Function is exact at given grid points $\{x_i\}$

$$f(x_j) = \sum_{i=0}^{N} c_i^{(N)} \phi_i(x_j)$$

- **Grid:** Lobatto (include end points), Gauss (exclude end points), Radau (include one end and excludes other)

- **Error / Chebyshev coefficients:** exponencial/algebraic convergence depending whether the function is analytic or $\mathcal{C}^{\ell}$

OUTLINE PART 2

- Derivatives
- Example 1: eigenvalue problems (Quasi-normal modes)
- Algorithm for solution of ordinary differential equations
- Example 2: Laplace equation
- Extension to partial differential equations

- Example 3: elliptic equation
  (rotating disk of charged dust in general relativity)
- Example 4: hyperbolic equation
  (conformal Einstein's field equations, linear problem)

DERIVATIVES

- Interpolate at any point ξ : $$\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} T_i(\xi)$$
- The derivative reads: $$\frac{d}{d\xi}\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} \frac{d}{d\xi} T_i(\xi)$$

DERIVATIVES

- Interpolate at any point ξ : $\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} T_i(\xi)$

- The derivative reads: $\frac{d}{d\xi}\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} \frac{d}{d\xi} T_i(\xi)$

- In particular, the derivative of a Chebyshev Polynomial can be expressed in terms of the Chebyshev basis

$$\frac{d}{d\xi} T_i(\xi) = \sum_{j=0}^{N} d_{ij}^{(N)} T_j(\xi)$$

DERIVATIVES

- Interpolate at any point ξ : $$\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} T_i(\xi)$$

- The derivative reads: $$\frac{d}{d\xi}\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} \frac{d}{d\xi} T_i(\xi)$$

- In particular, the derivative of a Chebyshev Polynomial can be expressed in terms of the Chebyshev basis

$$\frac{d}{d\xi} T_i(\xi) = \sum_{j=0}^{N} d_{ij}^{(N)} T_j(\xi)$$

are calculated exactly using properties of the Chebyshev Polynomials

DERIVATIVES

- Interpolate at any point ξ : $$\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} T_i(\xi)$$

- The derivative reads: $$\frac{d}{d\xi}\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} \frac{d}{d\xi} T_i(\xi)$$

- In particular, the derivative of a Chebyshev Polynomial can be expressed in terms of the Chebyshev basis

$$\frac{d}{d\xi} T_i(\xi) = \sum_{j=0}^{N} d_{ij}^{(N)} T_j(\xi)$$

$$\Downarrow$$

$$\frac{d}{d\xi}\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} \sum_{j=0}^{N} d_{ij}^{(N)} T_j(\xi)$$

DERIVATIVES

- Interpolate at any point ξ : $\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} T_i(\xi)$

- The derivative reads: $\frac{d}{d\xi}\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} \frac{d}{d\xi} T_i(\xi)$

- In particular, the derivative of a Chebyshev Polynomial can be expressed in terms of the Chebyshev basis

$$\frac{d}{d\xi} T_i(\xi) = \sum_{j=0}^{N} d_{ij}^{(N)} T_j(\xi)$$

$$\downarrow$$

$$\frac{d}{d\xi}\tilde{f}(\xi) = \sum_{j=0}^{N} \underbrace{\sum_{i=0}^{N} c_i^{(N)} d_{ij}^{(N)}}_{c'_j} T_j(\xi)$$

DERIVATIVES

- Interpolate at any point ξ : $$\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} T_i(\xi)$$
- The derivative reads: $$\frac{d}{d\xi}\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} \frac{d}{d\xi} T_i(\xi)$$

DERIVATIVES

- Interpolate at any point ξ : $$\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} T_i(\xi)$$
- The derivative reads: $$\frac{d}{d\xi}\tilde{f}(\xi) = \sum_{i=0}^{N} c_i'^{(N)} T_i(\xi)$$

DERIVATIVES

- Interpolate at any point ξ : $$\tilde{f}(\xi) = \sum_{i=0}^{N} c_i^{(N)} T_i(\xi)$$
- The derivative reads: $$\frac{d}{d\xi}\tilde{f}(\xi) = \sum_{i=0}^{N} c_i'^{(N)} T_i(\xi)$$

Derivative Coefficients algorithm

- Given resolution N and Chebyshev Coefficients $c_i^{(N)}$
- Recurrence relation for derivative coefficients $c'^{(N)}_i$
- Start with $c'^{(N)}_{N+1} = c'^{(N)}_N = 0$
- Run backward for $i = N \cdots 1$

$$c'_{i-1} = 2kc_i + c'_{k+1}$$

DISCRETE DERIVATIVES

- Consider the derivative at the grid points $\{\xi_k\}$

$$\frac{d}{d\xi}\tilde{f}(\xi_k) = \sum_{j=0}^{N}\sum_{i=0}^{N} c_i^{(N)} d_{ij}^{(N)} T_j(\xi_k)$$

DISCRETE DERIVATIVES

- Consider the derivative at the grid points $\{\xi_k\}$

$$\underbrace{\frac{d}{d\xi}\tilde{f}(\xi_k)}_{f'_k} = \sum_{j=0}^{N}\sum_{i=0}^{N} c_i^{(N)} d_{ij}^{(N)} \underbrace{T_j(\xi_k)}_{T_{kj}}$$

- Consider relation to find coefficients $\vec{c} = \hat{T}^{-1}\vec{f}$

DISCRETE DERIVATIVES

- Consider the derivative at the grid points $\{\xi_k\}$

$$\underbrace{\frac{d}{d\xi}\tilde{f}(\xi_k)}_{f'_k} = \sum_{j=0}^{N}\sum_{i=0}^{N} c_i^{(N)} d_{ij}^{(N)} \underbrace{T_j(\xi_k)}_{T_{kj}}$$

- Consider relation to find coefficients $c_i^{(N)} = \sum_{\ell=0}^{N} (T^{-1})_{i\ell} f_\ell$

$$f'_k = \sum_{j=0}^{N}\sum_{i=0}^{N}\sum_{\ell=0}^{N} (T^{-1})_{i\ell} f_\ell d_{ij}^{(N)} T_{kj}$$

DISCRETE DERIVATIVES

- Consider the derivative at the grid points $\{\xi_k\}$

$$\underbrace{\frac{d}{d\xi}\tilde{f}(\xi_k)}_{f'_k} = \sum_{j=0}^{N}\sum_{i=0}^{N} c_i^{(N)} d_{ij}^{(N)} \underbrace{T_j(\xi_k)}_{T_{kj}}$$

- Consider relation to find coefficients $c_i^{(N)} = \sum_{\ell=0}^{N} (T^{-1})_{i\ell} f_\ell$

$$f'_k = \sum_{\ell=0}^{N} \underbrace{\sum_{i=0}^{N}\sum_{j=0}^{N} T_{kj} d_{ij}^{(N)} (T^{-1})_{i\ell}}_{D_{k\ell}} f_\ell$$

DISCRETE DERIVATIVES

- Consider the derivative at the grid points $\{\xi_k\}$

$$\underbrace{\frac{d}{d\xi}\tilde{f}(\xi_k)}_{f'_k} = \sum_{j=0}^{N}\sum_{i=0}^{N} c_i^{(N)} d_{ij}^{(N)} \underbrace{T_j(\xi_k)}_{T_{kj}}$$

- Consider relation to find coefficients $c_i^{(N)} = \sum_{\ell=0}^{N} (T^{-1})_{i\ell} f_\ell$

$$f'_k = \sum_{\ell=0}^{N} D_{k\ell} f_\ell \Rightarrow \vec{f}' = \hat{D}\vec{f}$$

- Spectral differentiation matrices: $\hat{D}$

DISCRETE DERIVATIVES

- Continuous derivative operator: ∂_μ
- Discrete derivative operator: $\hat{D}_\mu$

DISCRETE DERIVATIVES

- Continuous derivative operator: ∂_μ
- Discrete derivative operator: $\hat{D}_\mu$
    - ➡ Dense matrix

DISCRETE DERIVATIVES

- Continuous derivative operator: ∂_μ
- Discrete derivative operator: $\hat{D}_\mu$

➡ Dense matrix

➡ Eventually one needs to invert such matrices: numerically expansive procedure

DISCRETE DERIVATIVES

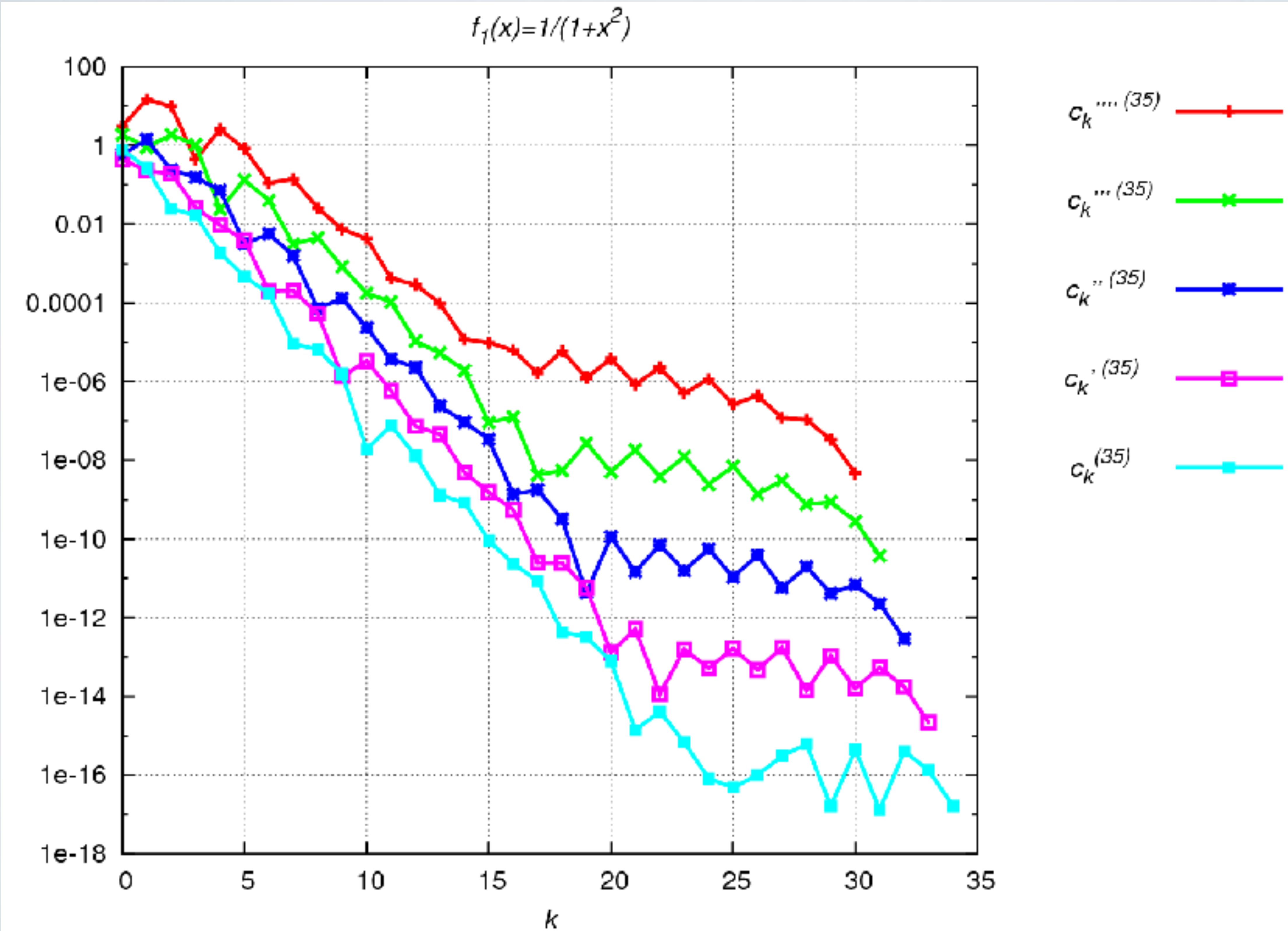
- Continuous derivative operator: ∂_μ
- Discrete derivative operator: $\hat{D}_\mu$

➡ Dense matrix

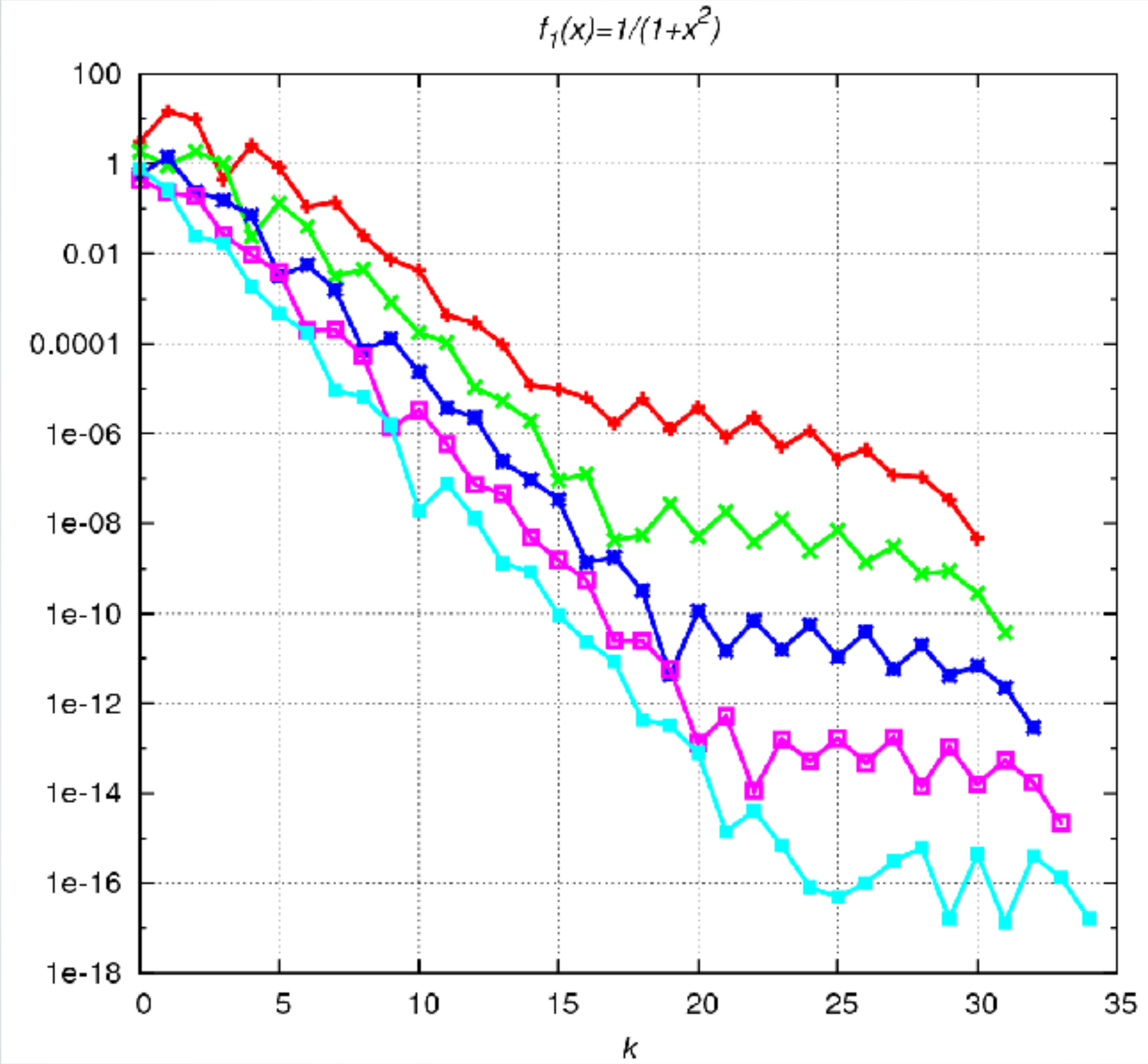
➡ Eventually one needs to invert such matrices: numerically expansive procedure

➡ Contains a lot of information (see example eigenvalue problem)

DISCRETE DERIVATIVES



DISCRETE DERIVATIVES



For each derivative, approximately two digits of accuracy are lost

EXAMPLE 1: QNM

- **Perturbation Theory: typical problem**

- Stationary (black-hole) spacetime as background
- Consider fields propagating on background
- Linearise or consider a linear theory
- Wave equation
- Take boundary conditions into account
- Fourier (or Laplace) transformation
- Non-trivial solutions to the homogenous equation (eigenvalue problem)

EXAMPLE 1: QNM

- **Perturbation Theory: typical problem**

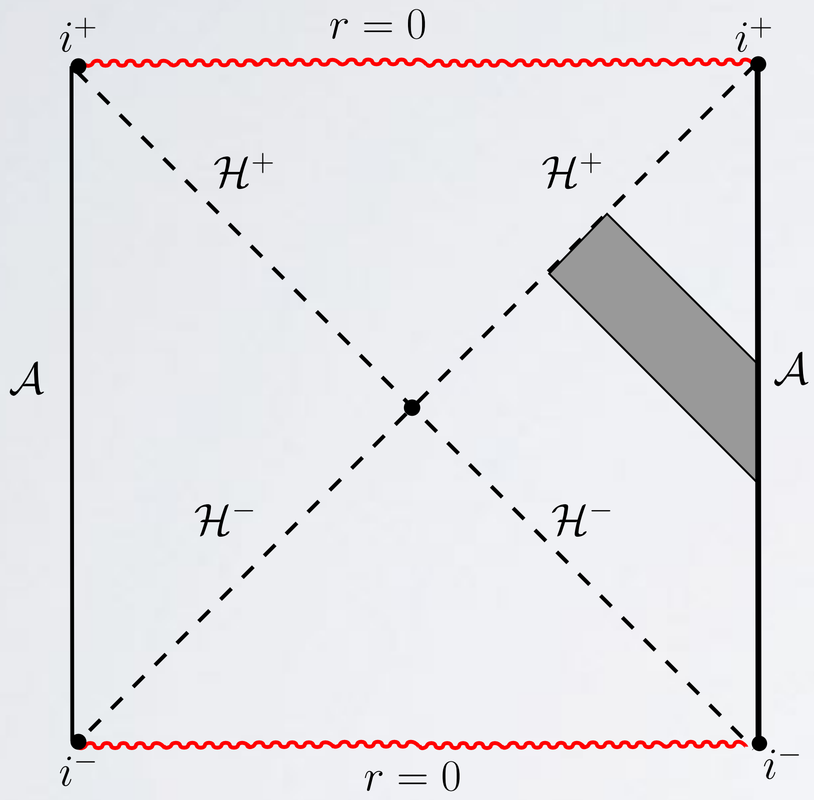
- Stationary (black-hole) spacetime as background
- Consider fields propagating on background
- Linearise or consider a linear theory
- Wave equation
- Take boundary conditions into account
- Fourier (or Laplace) transformation
- Non-trivial solutions to the homogenous equation (eigenvalue problem)

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Black brane background spacetime:** Schwarzschild-AdS in 5D

$$ds^2 = \frac{1}{\rho^2}\left(-f(\rho)dv^2 - 2\,dvd\rho + dx^2 + dy^2 + dz^2\right) \qquad f(\rho) = 1 - \rho^4$$



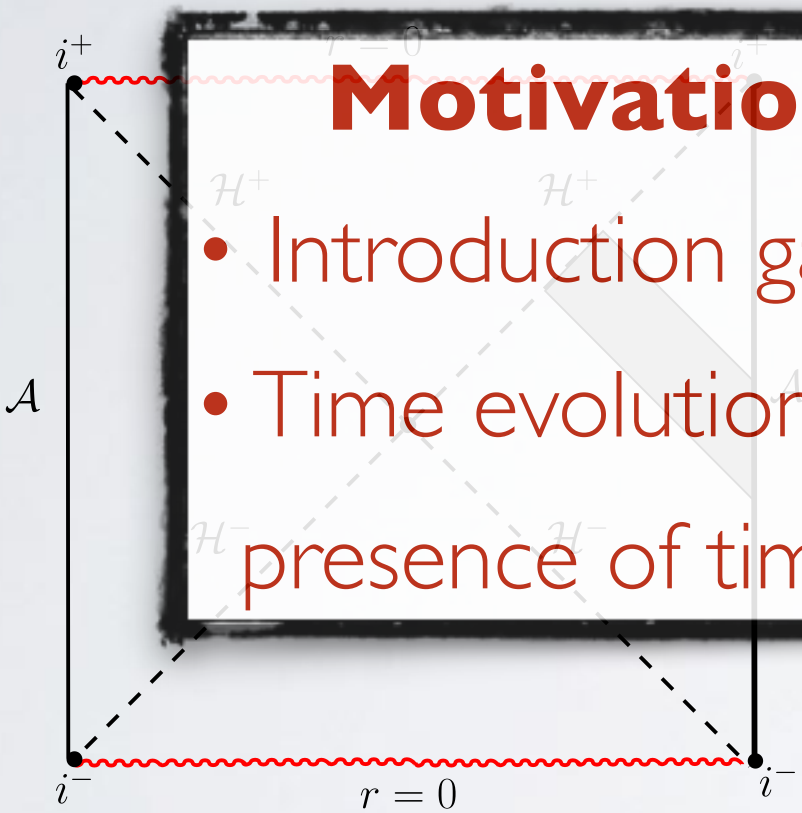
- *Coordinates:* Ingoing Eddington-Finkelstein (horizon penetrating) $\quad \rho = 1$
- *Null Infinity:* AdS boundary (time-like surface) $\quad \rho = 0$

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Black brane background spacetime:** Schwarzschild-AdS in 5D

$$ds^2 = \frac{1}{\rho^2}\left(-f(\rho)dv^2 - 2\,dvd\rho + dx^2 + dy^2 + dz^2\right) \qquad f(\rho) = 1 - \rho^4$$



Motivation AdS/CFT Correspondence

- Introduction gauge/gravity duality (O. Miskovic)
- Time evolution of the chiral magnetic effect in the presence of time dependent axial charges

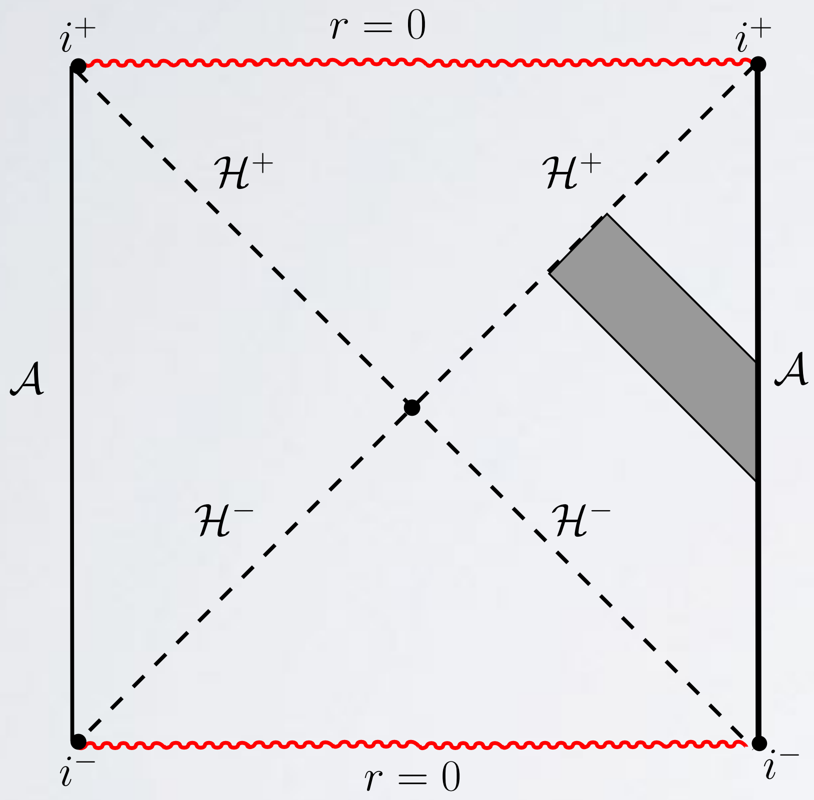
- Coordinates: Ingoing Eddington-Finkelstein (horizon penetrating) $\rho = 1$
- Null Infinity: AdS boundary (time-like surface) $\rho = 0$

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Black brane background spacetime:** Schwarzschild-AdS in 5D

$$ds^2 = \frac{1}{\rho^2}\left(-f(\rho)dv^2 - 2\,dv d\rho + dx^2 + dy^2 + dz^2\right) \qquad f(\rho) = 1 - \rho^4$$



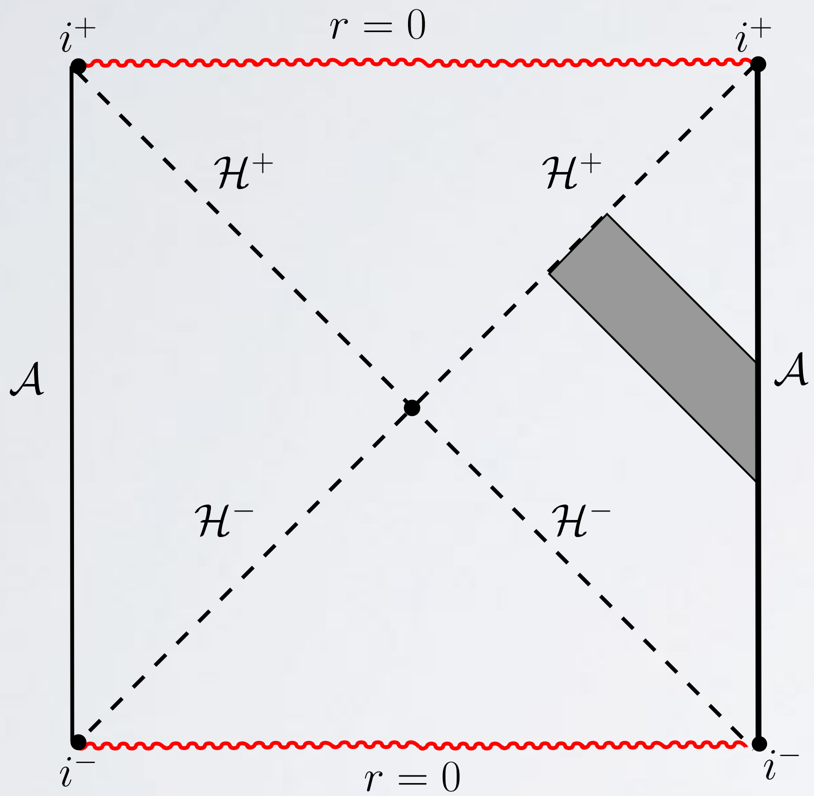
- *Coordinates:* Ingoing Eddington-Finkelstein (horizon penetrating) $\rho = 1$
- *Null Infinity:* AdS boundary (time-like surface) $\rho = 0$

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Black brane background spacetime:** Schwarzschild-AdS in 5D

$$ds^2 = \frac{1}{\rho^2}\left(-f(\rho)dv^2 - 2\,dv d\rho + dx^2 + dy^2 + dz^2\right) \qquad f(\rho) = 1 - \rho^4$$



- *Coordinates:* Ingoing Eddington-Finkelstein (horizon penetrating) $\rho = 1$
- *Null Infinity:* AdS boundary (time-like surface) $\rho = 0$

- **Wave equation:**

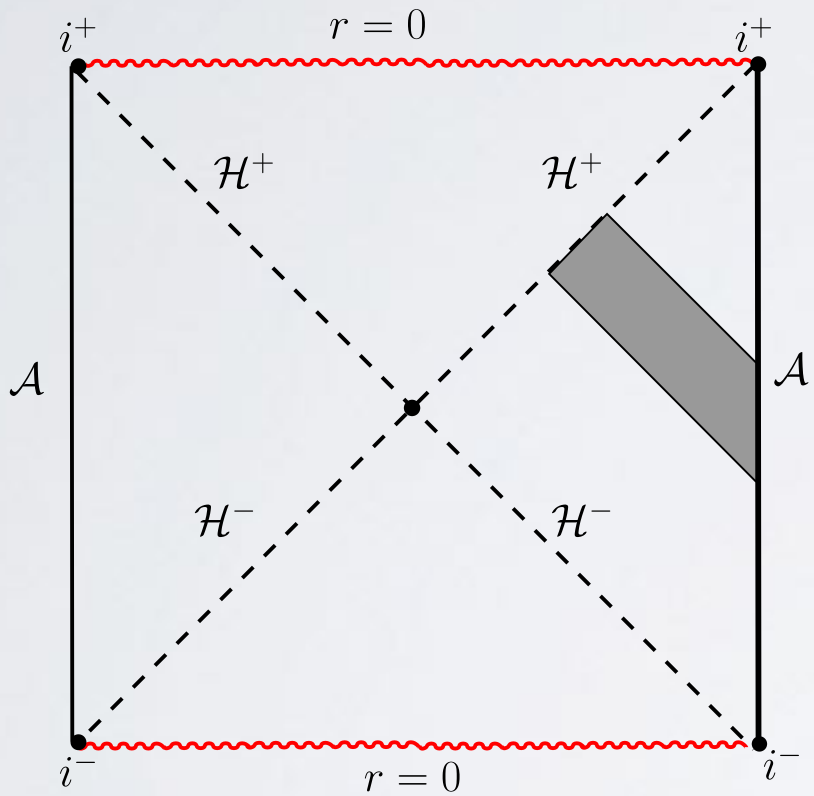
$$\left[-\rho\,(1-\rho^4)\,\frac{\partial^2}{\partial\rho^2} - (3-7\rho^4)\,\frac{\partial}{\partial\rho} + (8+\lambda^2)\,\rho^3\right]U(v,\rho) + \left[2\,\rho\,\frac{\partial}{\partial\rho} + 3\right]\dot{U}(v,\rho) + S(v,\rho) = 0.$$

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Black brane background spacetime:** Schwarzschild-AdS in 5D

$$ds^2 = \frac{1}{\rho^2}\left(-f(\rho)dv^2 - 2\,dvd\rho + dx^2 + dy^2 + dz^2\right) \qquad f(\rho) = 1-\rho^4$$



- *Coordinates:* Ingoing Eddington-Finkelstein (horizon penetrating) $\rho = 1$
- *Null Infinity:* AdS boundary (time-like surface) $\rho = 0$

- **Wave equation:**

$$\left[-\rho\,(1-\rho^4)\,\frac{\partial^2}{\partial\rho^2} - (3-7\rho^4)\,\frac{\partial}{\partial\rho} + (8+\lambda^2)\,\rho^3\right]U(v,\rho) + \left[2\,\rho\,\frac{\partial}{\partial\rho} + 3\right]\dot{U}(v,\rho) + S(v,\rho) = 0.$$

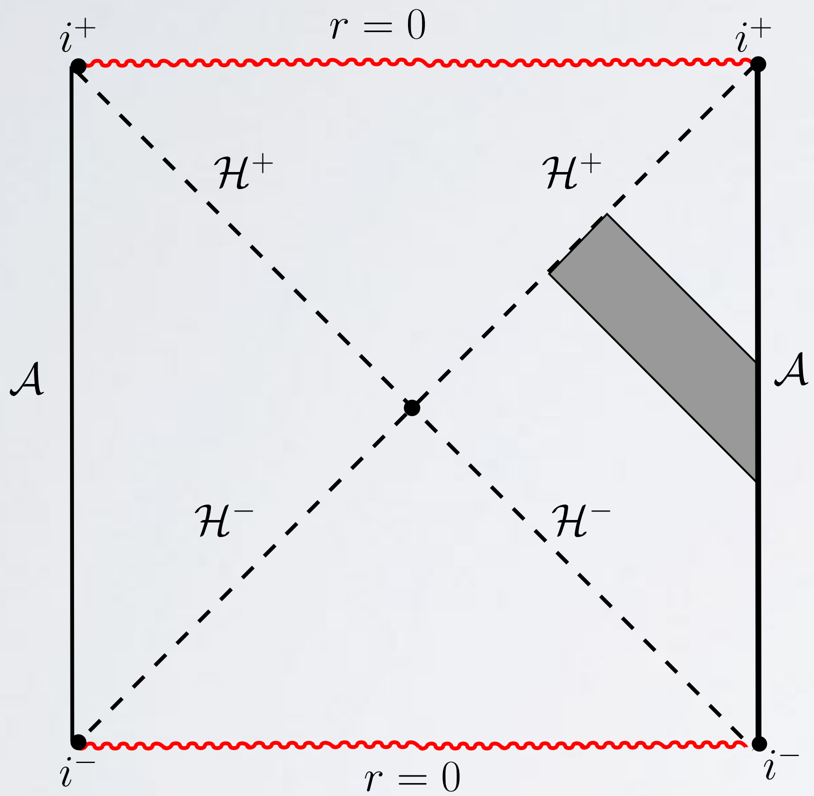
time-derivative

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Black brane background spacetime:** Schwarzschild-AdS in 5D

$$ds^2 = \frac{1}{\rho^2}\left(-f(\rho)dv^2 - 2\,dvd\rho + dx^2 + dy^2 + dz^2\right) \qquad f(\rho) = 1-\rho^4$$



- *Coordinates:* Ingoing Eddington-Finkelstein (horizon penetrating) $\quad \rho = 1$
- *Null Infinity:* AdS boundary (time-like surface) $\quad \rho = 0$

- **Wave equation:**

$$\left[-\rho\,(1-\rho^4)\,\frac{\partial^2}{\partial\rho^2} - (3-7\rho^4)\,\frac{\partial}{\partial\rho} + (8+\lambda^2)\,\rho^3\right] U(v,\rho) + \left[2\,\rho\,\frac{\partial}{\partial\rho} + 3\right]\dot{U}(v,\rho) + S(v,\rho) = 0.$$

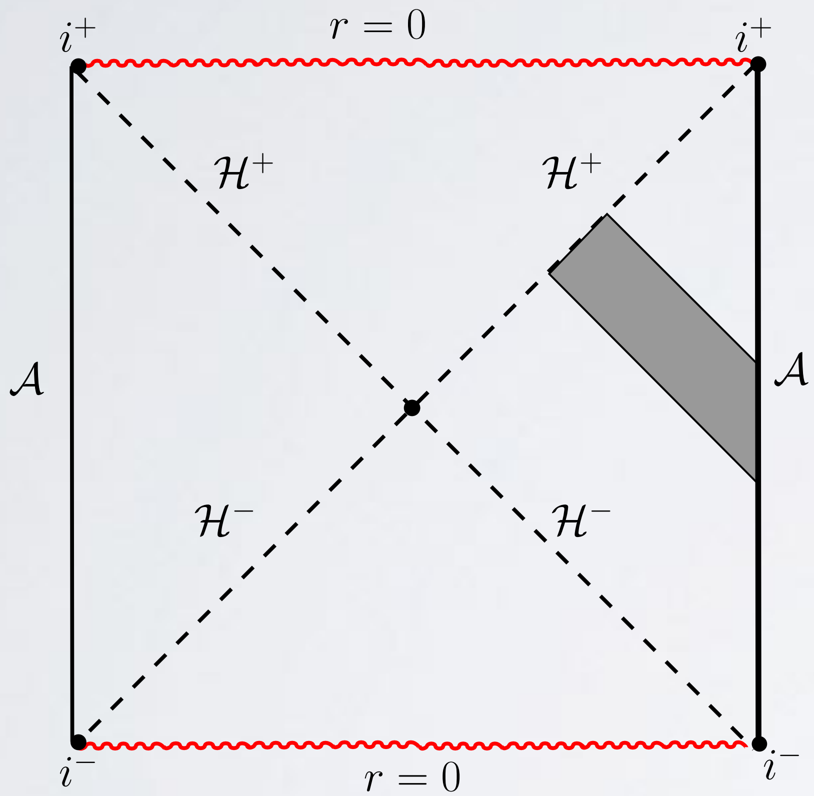
time-derivative second order

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Black brane background spacetime:** Schwarzschild-AdS in 5D

$$ds^2 = \frac{1}{\rho^2}\left(-f(\rho)dv^2 - 2\,dvd\rho + dx^2 + dy^2 + dz^2\right) \qquad f(\rho) = 1 - \rho^4$$



- *Coordinates:* Ingoing Eddington-Finkelstein (horizon penetrating) $\rho = 1$
- *Null Infinity:* AdS boundary (time-like surface) $\rho = 0$

- **Laplace transformation (homogenous equation):**

$$\left[-\rho\,(1-\rho^4)\,\frac{\partial^2}{\partial\rho^2} - \left(3 - 7\rho^4\right)\,\frac{\partial}{\partial\rho} + (8+\lambda^2)\,\rho^3\right]\bar{U}(\rho) + s\left[2\,\rho\,\frac{\partial}{\partial\rho} + 3\,\right]\bar{U}(\rho) = 0.$$

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Problem:** find complex s-values for which equation has a non-vanishing regular solution

$$\left[-\rho\,(1-\rho^4)\,\frac{\partial^2}{\partial\rho^2}-\left(3-7\rho^4\right)\,\frac{\partial}{\partial\rho}+(8+\lambda^2)\,\rho^3\right]\bar{U}(\rho)+{\color{red}s}\left[2\,\rho\,\frac{\partial}{\partial\rho}+3\,\right]\bar{U}(\rho)=0.$$

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Problem:** find complex s-values for which equation has a non-vanishing regular solution

$$\left[-\rho\,(1-\rho^4)\,\frac{\partial^2}{\partial\rho^2}-\left(3-7\rho^4\right)\,\frac{\partial}{\partial\rho}+(8+\lambda^2)\,\rho^3\right]\bar{U}(\rho)+{\color{red}s}\left[2\,\rho\,\frac{\partial}{\partial\rho}+3\,\right]\bar{U}(\rho)=0.$$

- **Numerical solution with spectral methods:**

1. Discretise domain $\rho\in[0,1]$ with Lobatto-grid (include end points) leads to $\bar{U}(\rho)\to\vec{U}$ and $\partial_\rho\to\hat{D}$

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Problem:** find complex s-values for which equation has a non-vanishing regular solution

$$\left[-\rho\,(1-\rho^4)\,\frac{\partial^2}{\partial\rho^2}-\left(3-7\rho^4\right)\,\frac{\partial}{\partial\rho}+(8+\lambda^2)\,\rho^3\right]\bar{U}(\rho)+s\left[2\,\rho\,\frac{\partial}{\partial\rho}+3\right]\bar{U}(\rho)=0.$$

- **Numerical solution with spectral methods:**

1. Discretise domain $\rho\in[0,1]$ with Lobatto-grid (include end points) leads to $\bar{U}(\rho)\to\vec{U}$ and $\partial_\rho\to\hat{D}$

$$\left[-\rho\,(1-\rho^4)\,\hat{D}\cdot\hat{D}-\left(3-7\rho^4\right)\,\hat{D}+(8+\lambda^2)\,\rho^3\cdot\mathbb{1}\right]\vec{U}+s\left[2\,\rho\,\hat{D}+3\cdot\mathbb{1}\right]\vec{U}=0.$$

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Problem:** find complex s-values for which equation has a non-vanishing regular solution

$$\left[-\rho\,(1-\rho^4)\,\frac{\partial^2}{\partial\rho^2} - \left(3-7\rho^4\right)\,\frac{\partial}{\partial\rho} + (8+\lambda^2)\,\rho^3\right]\bar{U}(\rho) + s\left[2\,\rho\,\frac{\partial}{\partial\rho} + 3\,\right]\bar{U}(\rho) = 0.$$

- **Numerical solution with spectral methods:**

1. Discretise domain $\rho \in [0,1]$ with Lobatto-grid (include end points) leads to $\bar{U}(\rho) \to \vec{U}$ and $\partial_\rho \to \hat{D}$

$$\hat{\alpha}\vec{U} + s\hat{\beta}\vec{U} = 0.$$

EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Problem:** find complex s-values for which equation has a non-vanishing regular solution

$$\left[-\rho\,(1-\rho^4)\,\frac{\partial^2}{\partial\rho^2}-\left(3-7\rho^4\right)\,\frac{\partial}{\partial\rho}+(8+\lambda^2)\,\rho^3\right]\bar{U}(\rho)+s\left[2\,\rho\,\frac{\partial}{\partial\rho}+3\,\right]\bar{U}(\rho)=0.$$

- **Numerical solution with spectral methods:**

1. Discretise domain $\rho\in[0,1]$ with Lobatto-grid (include end points) leads to $\bar{U}(\rho)\rightarrow\vec{U}$ and $\partial_\rho\rightarrow\hat{D}$

$$\hat{\alpha}\vec{U}+s\hat{\beta}\vec{U}=0.$$

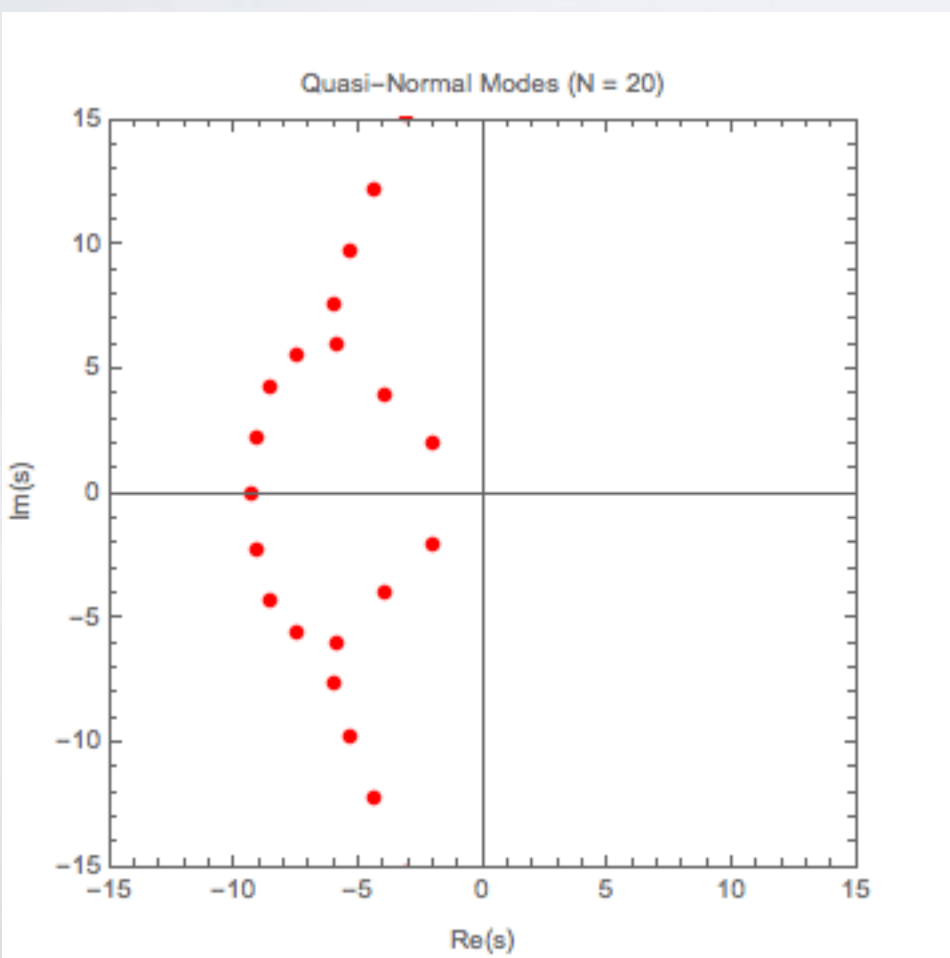
2. Ask your favourite mathematical software to solve the (generalised) eigenvalue problem

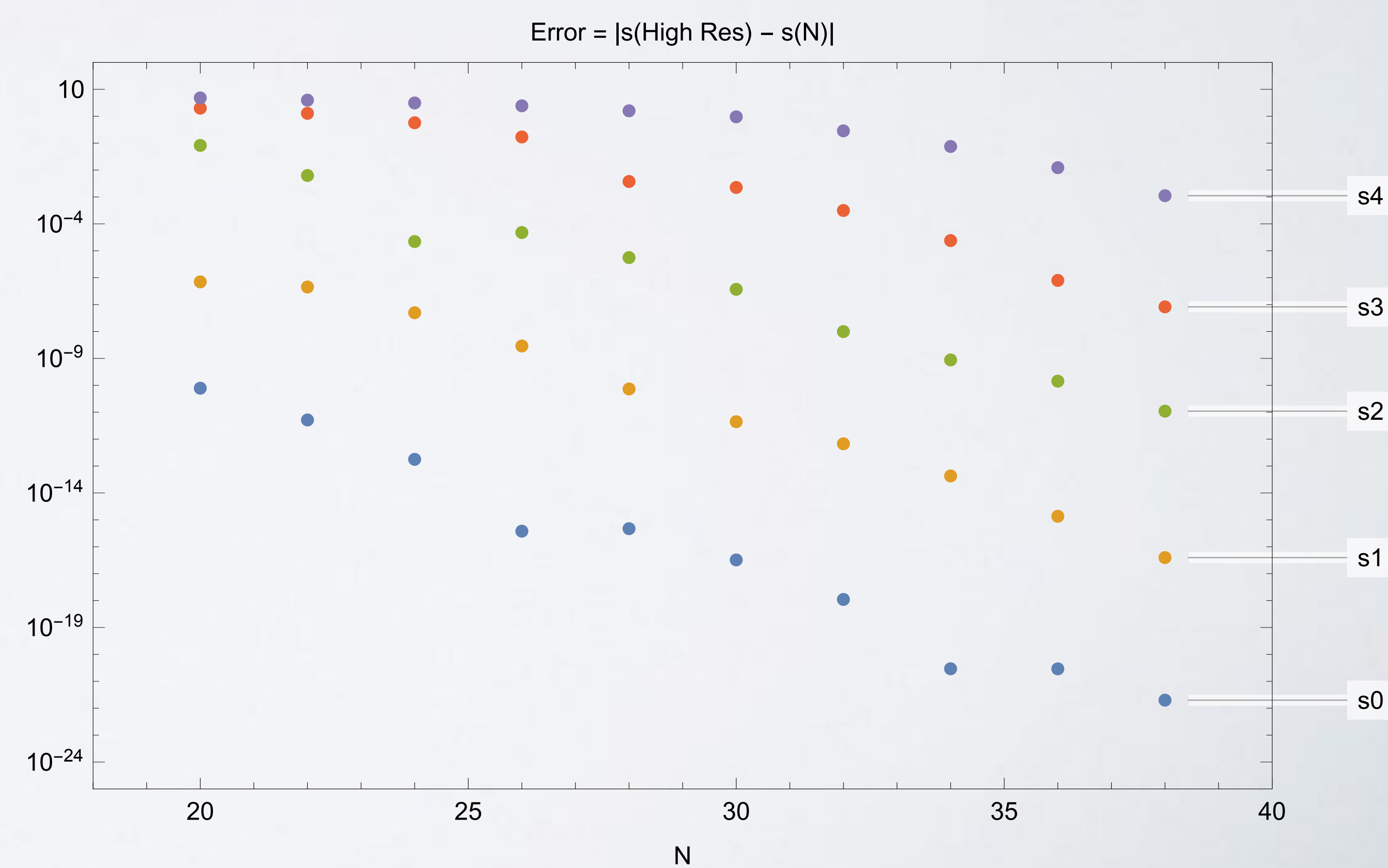
EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Remarks:**

N N

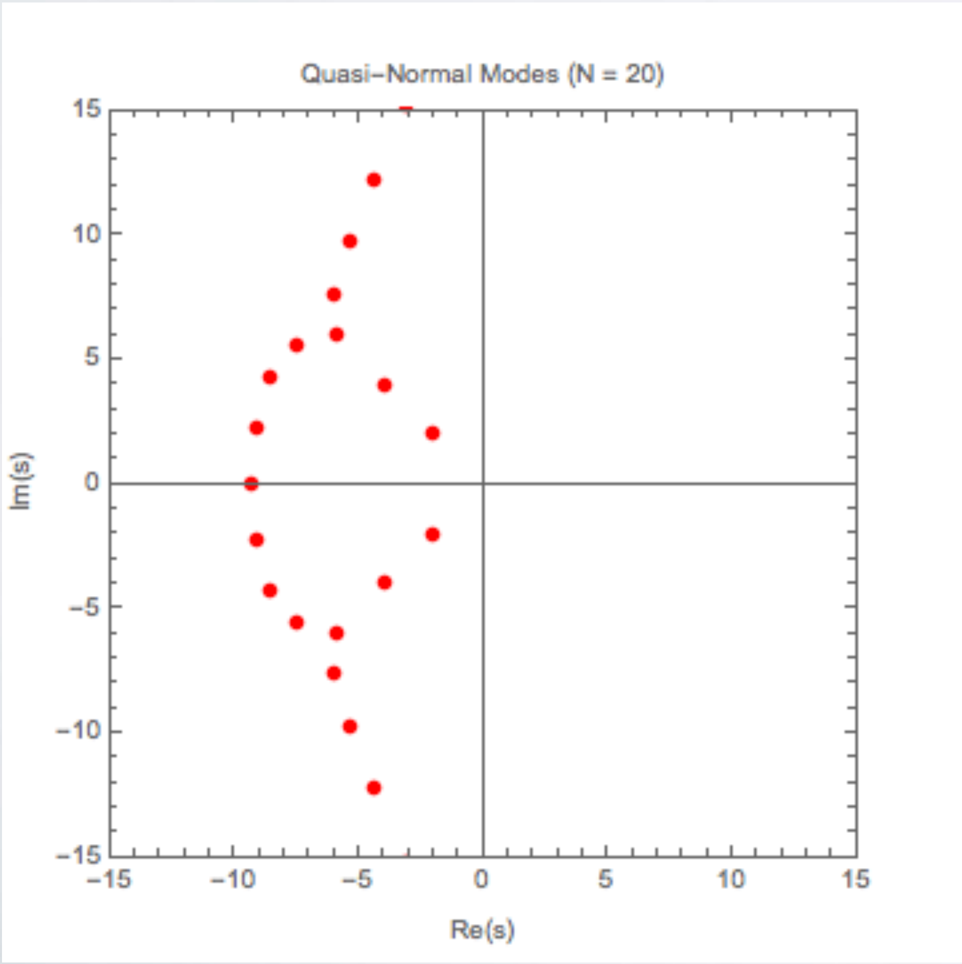


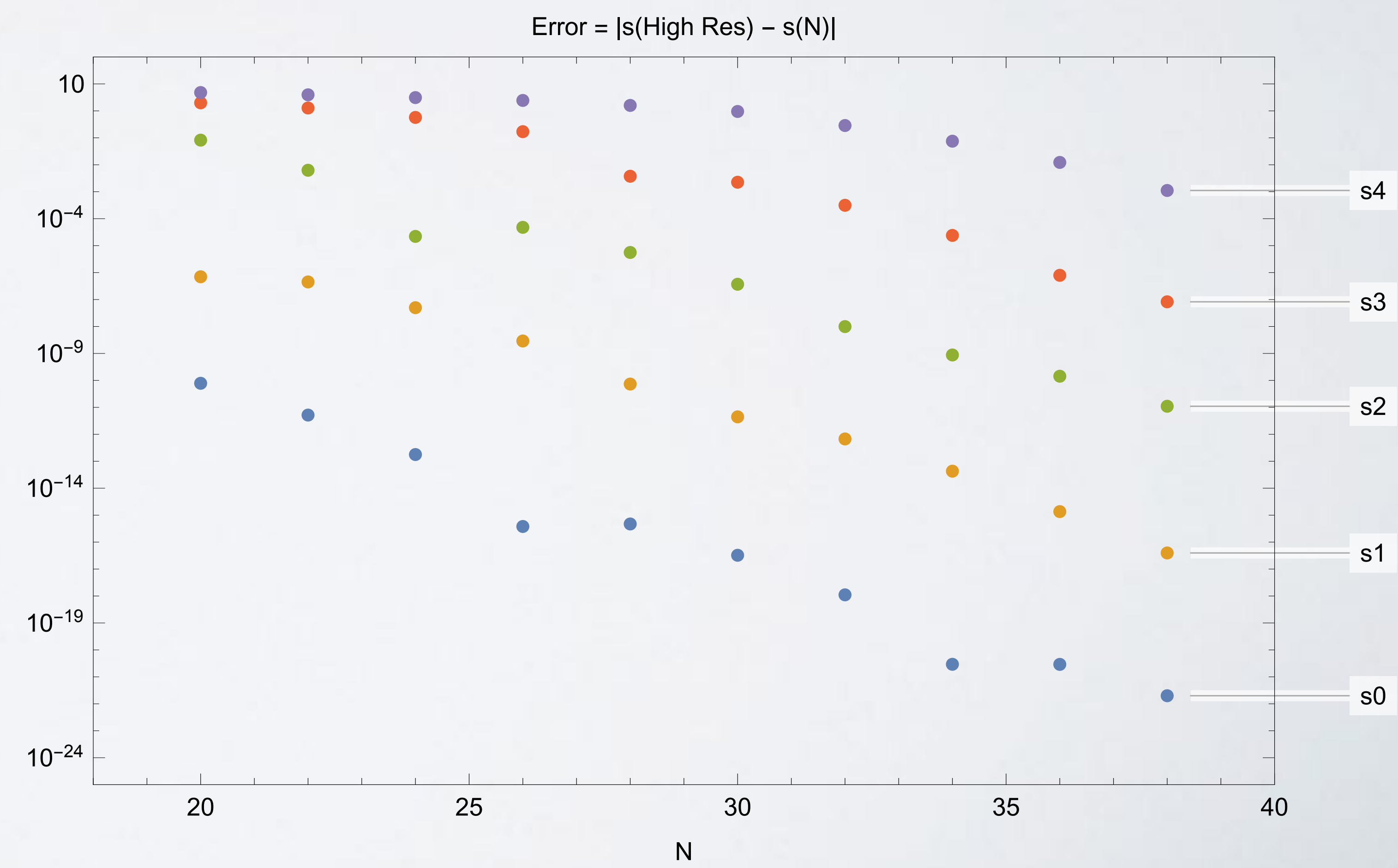


EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Remarks:**
  1. Given numerical resolution N leads to N "eigenvalues"
  2. However, most of them are rubbish
  3. One must study the convergence to see which values are stable and converge to a fixed value as we increase the resolution

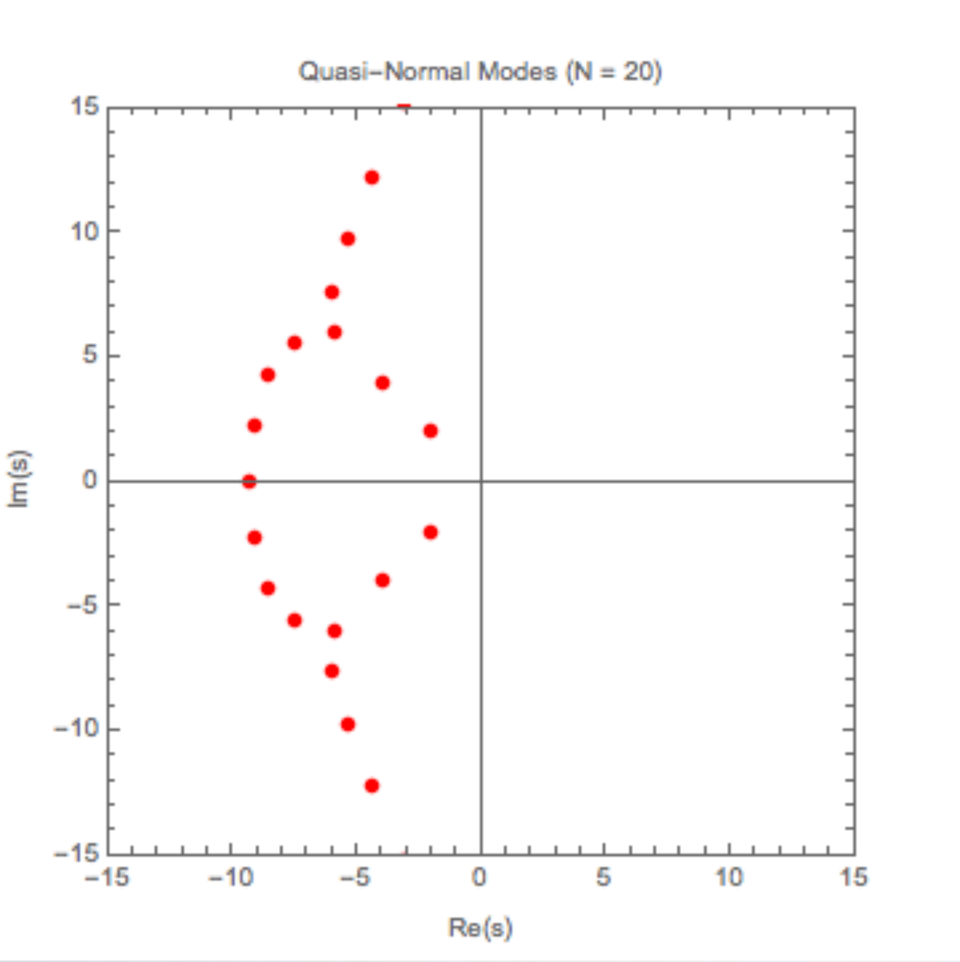


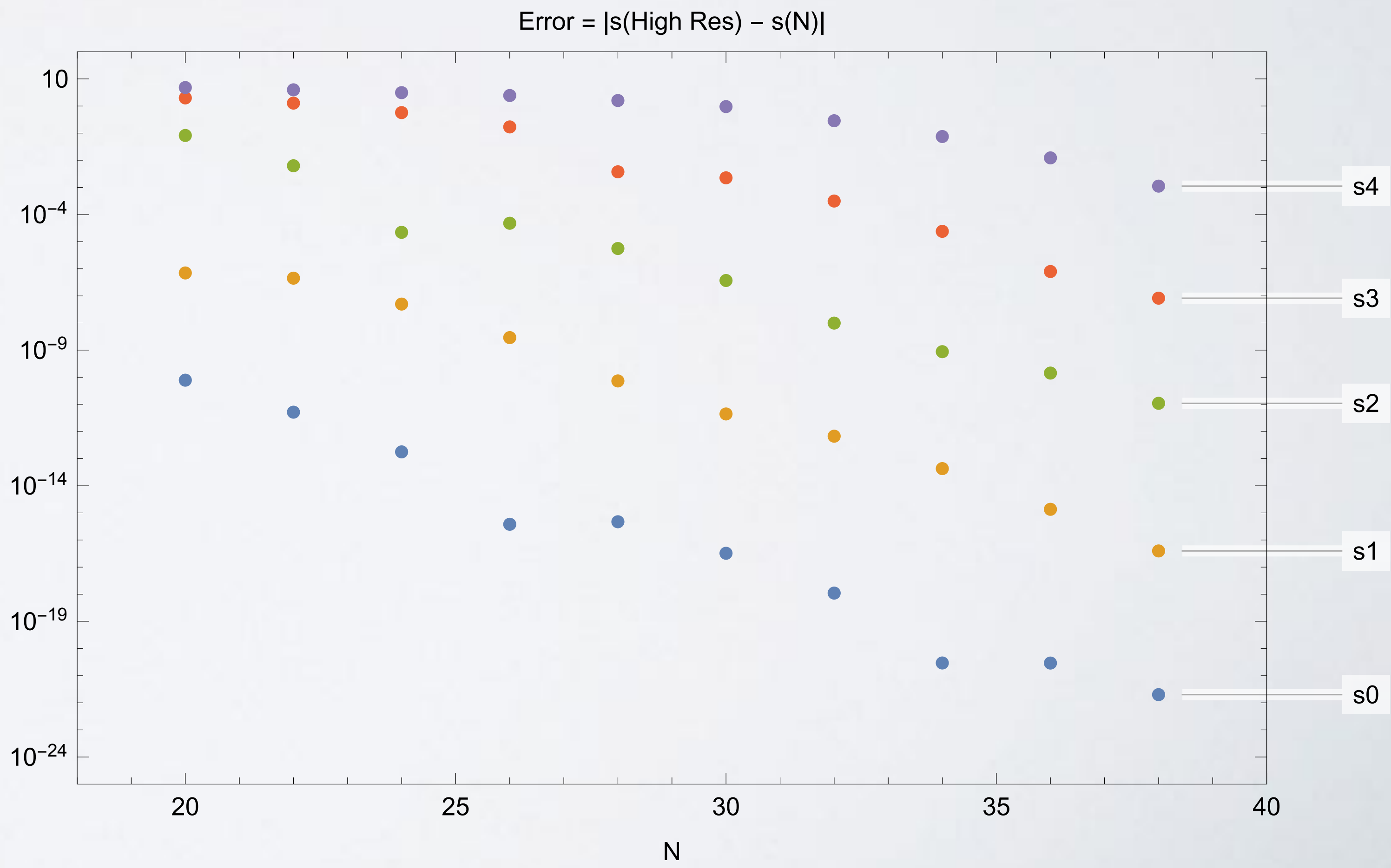


EXAMPLE 1: QNM

M. AMMON, S.GRIENINGER, A.J. ALBA, RPM, L. MELGAR, JHEP 09 (2016) 131

- **Remarks:**
  1. Given numerical resolution N leads to N "eigenvalues"
  2. However, most of them are rubbish
  3. One must study the convergence to see which values are stable and converge to a fixed value as we increase the resolution





SOLUTION ALGORITHM

- **Problem:** find solution for a function $f(x)$ with $x \in [a, b]$ which satisfies a given ordinary differential equation subject to boundary conditions (BC)
- **Extra issues:** sometimes the problem involves extra unknown parameters, (indirectly) subject to subsidiary conditions

SOLUTION ALGORITHM

- **Problem:** find solution for a function $f(x)$ with $x \in [a,b]$ which satisfies a given ordinary differential equation subject to boundary conditions (BC)
- **Extra issues:** sometimes the problem involves extra unknown parameters, (indirectly) subject to subsidiary conditions
- **Example:** Legendre equation

$$(1-x^2)f'' - 2xf' + \lambda f = 0 \qquad \begin{matrix} x \in [-1,1] \\ (x = \cos\theta) \end{matrix}$$

SOLUTION ALGORITHM

- **Problem:** find solution for a function $f(x)$ with $x \in [a, b]$ which satisfies a given ordinary differential equation subject to boundary conditions (BC)
- **Extra issues:** sometimes the problem involves extra unknown parameters, (indirectly) subject to subsidiary conditions
- **Example:** Legendre equation

$$(1 - x^2) f'' - 2x f' + \lambda f = 0 \qquad \begin{array}{c} x \in [-1, 1] \\ (x = \cos\theta) \end{array}$$

BC: regularity condition

$$2 f'_{|-1} + \lambda f_{|-1} = 0$$

$$-2 f'_{|1} + \lambda f_{|1} = 0$$

SOLUTION ALGORITHM

- **Problem:** find solution for a function $f(x)$ with $x \in [a,b]$ which satisfies a given ordinary differential equation subject to boundary conditions (BC)
- **Extra issues:** sometimes the problem involves extra unknown parameters, (indirectly) subject to subsidiary conditions
- **Example:** Legendre equation

$$(1-x^2)f'' - 2xf' + \lambda f = 0 \qquad \begin{array}{c} x \in [-1,1] \\ (x = \cos\theta) \end{array}$$

BC: regularity condition

$$2f'_{|-1} + \lambda f_{|-1} = 0$$

$$-2f'_{|1} + \lambda f_{|1} = 0$$

extra unknown parameter λ

SOLUTION ALGORITHM

- **Problem:** find solution for a function $f(x)$ with $x \in [a, b]$ which satisfies a given ordinary differential equation subject to boundary conditions (BC)
- **Extra issues:** sometimes the problem involves extra unknown parameters, (indirectly) subject to subsidiary conditions
- **Example:** Legendre equation

$$(1 - x^2)f'' - 2xf' + \lambda f = 0 \qquad \begin{matrix} x \in [-1, 1] \\ (x = \cos\theta) \end{matrix}$$

BC: regularity condition

$$2f'_{|-1} + \lambda f_{|-1} = 0$$

$$-2f'_{|1} + \lambda f_{|1} = 0$$

extra unknown parameter λ

subsidiary condition: normalisation

$$f(1) = 1$$

SOLUTION ALGORITHM

- **Notation:** consider the second order differential equation in the form

$$F(f, f', f''; x, \lambda) = 0$$

with boundary conditions

$$F_a(f(a), f'(a); \lambda) = 0 \qquad F_b(f(b), f'(b); \lambda) = 0$$

and eventual complementary condition

$$F_*(f(x_*), f'(x_*), f''(x_*); \lambda) = 0$$

SOLUTION ALGORITHM

- **Notation:** consider the second order differential equation in the form

$$F(f, f', f''; x, \lambda) = 0$$

with boundary conditions

$$F_a(f(a), f'(a); \lambda) = 0 \qquad F_b(f(b), f'(b); \lambda) = 0$$

and eventual complementary condition

$$F_*(f(x_*), f'(x_*), f''(x_*); \lambda) = 0$$

- **Numerical implementation:** fix resolution N and consider a vector with all unknowns:

$$\vec{X}^{\mathrm{T}} = (f_0 \cdots f_N | \lambda)$$

SOLUTION ALGORITHM

- **Notation:** consider the second order differential equation in the form

$$F(f, f', f''; x, \lambda) = 0$$

with boundary conditions

$$F_a(f(a), f'(a); \lambda) = 0 \qquad F_b(f(b), f'(b); \lambda) = 0$$

and eventual complementary condition

$$F_*(f(x_*), f'(x_*), f''(x_*); \lambda) = 0$$

- **Numerical implementation:** fix resolution N and consider a vector with all unknowns:

$$\vec{X}^{\mathrm{T}} = (f_0 \cdots f_N | \lambda)$$

function values at grid points

SOLUTION ALGORITHM

- **Notation:** consider the second order differential equation in the form

$$F(f, f', f''; x, \lambda) = 0$$

with boundary conditions

$$F_a(f(a), f'(a); \lambda) = 0 \qquad F_b(f(b), f'(b); \lambda) = 0$$

and eventual complementary condition

$$F_*(f(x_*), f'(x_*), f''(x_*); \lambda) = 0$$

- **Numerical implementation:** fix resolution N and consider a vector with all unknowns:

$$\vec{X}^{\mathrm{T}} = (f_0 \cdots f_N | \lambda)$$

function values at grid points

eventual extra parameters

SOLUTION ALGORITHM

- **Notation:** consider the second order differential equation in the form

$$F(f, f', f''; x, \lambda) = 0$$

with boundary conditions

$$F_a(f(a), f'(a); \lambda) = 0 \qquad F_b(f(b), f'(b); \lambda) = 0$$

and eventual complementary condition

$$F_*(f(x_*), f'(x_*), f''(x_*); \lambda) = 0$$

- **Numerical implementation:** fix resolution N and consider a vector with all unknowns:

$$\vec{X}^{\mathrm{T}} = (f_0 \cdots f_N | \lambda)$$

- **Dimension:** total number of unknown $n_{\text{total}} = n_{\text{fields}}(N+1) + n_{\text{par}}$

SOLUTION ALGORITHM

- **Notation:** consider the second order differential equation in the form

$$F(f, f', f''; x, \lambda) = 0$$

with boundary conditions

$$F_a(f(a), f'(a); \lambda) = 0 \qquad F_b(f(b), f'(b); \lambda) = 0$$

and eventual complementary condition

$$F_*(f(x_*), f'(x_*), f''(x_*); \lambda) = 0$$

- **Numerical implementation:** fix resolution N and consider a vector with all unknowns:

$$\vec{X}^{\mathrm{T}} = (f_0 \cdots f_N | \lambda)$$

From any such vector we can compute the spectral coefficients and the corresponding derivatives $\vec{f}, \vec{f'}, \vec{f''}$

SOLUTION ALGORITHM

- With $\vec{f}$, $\vec{f'}$, $\vec{f''}$, we evaluate the differential equation, boundary conditions and eventual subsidiary equations at the grid points and obtain the vector

$$\vec{F}^{\mathrm{T}} = (F_0 \cdots F_{\mathrm{Ntotal}})$$

whose components are

$$F_k = \begin{cases} F_a(f_0, f_0'; \lambda) & k = 0 \\ F(f_k, f_k', f_k''; \lambda) & 0 < k < N \\ F_b(f_N, f_N'; \lambda) & k = N \\ F_*(f_*, f_*'; \lambda) & k = N_{\mathrm{total}} \end{cases}$$

- The system correspond to an algebraic system

$$\vec{F}(\vec{X}) = 0$$

SOLUTION ALGORITHM

- **Example:** Legendre equation

$$(1-x^2)f'' - 2xf' + \lambda f = 0$$

SOLUTION ALGORITHM

- **Example:** Legendre equation

$$(1-x^2)f'' - 2xf' + \lambda f = 0$$

- Discretisation: Lobatto grid (end points included)

$$(1-x_k^2)f_k'' - 2x_k f_k' + \lambda f_k = 0$$

- Normalisation $f_0 - 1 = 0$

SOLUTION ALGORITHM

- **Example:** Legendre equation

$$(1-x^2)f'' - 2xf' + \lambda f = 0$$

- Discretisation: Lobatto grid (end points included)

$$(1-x_k^2)f_k'' - 2x_k f_k' + \lambda f_k = 0$$

- Normalisation $f_0 - 1 = 0$

- Here, the discrete algebraic system $\vec{F}(\vec{X})$ is formed as

$$F_k = \begin{cases} (1-x_k^2)\left(\hat{D}^2\vec{X}\right)_k - 2x_k\left(\hat{D}\vec{X}\right)_k + X_{N+1}X_k & 0 \le k \le N \\ X_0 - 1 & k = N+1 \end{cases}$$

SOLUTION ALGORITHM

- **Example:** Legendre equation

$$(1-x^2)f'' - 2xf' + \lambda f = 0$$

Linear X Non-Linear

(i) Originally, the Legendre equation is linear on the unknown function $f(x)$

(ii) However, we incorporate λ also as a variable

(iii) Coupling between λ and $f(x)$ leads to a non-linear equation

- Discretisation:

$$(1-x_k^2)f_k'' - 2x_k f_k' + \lambda f_k = 0$$

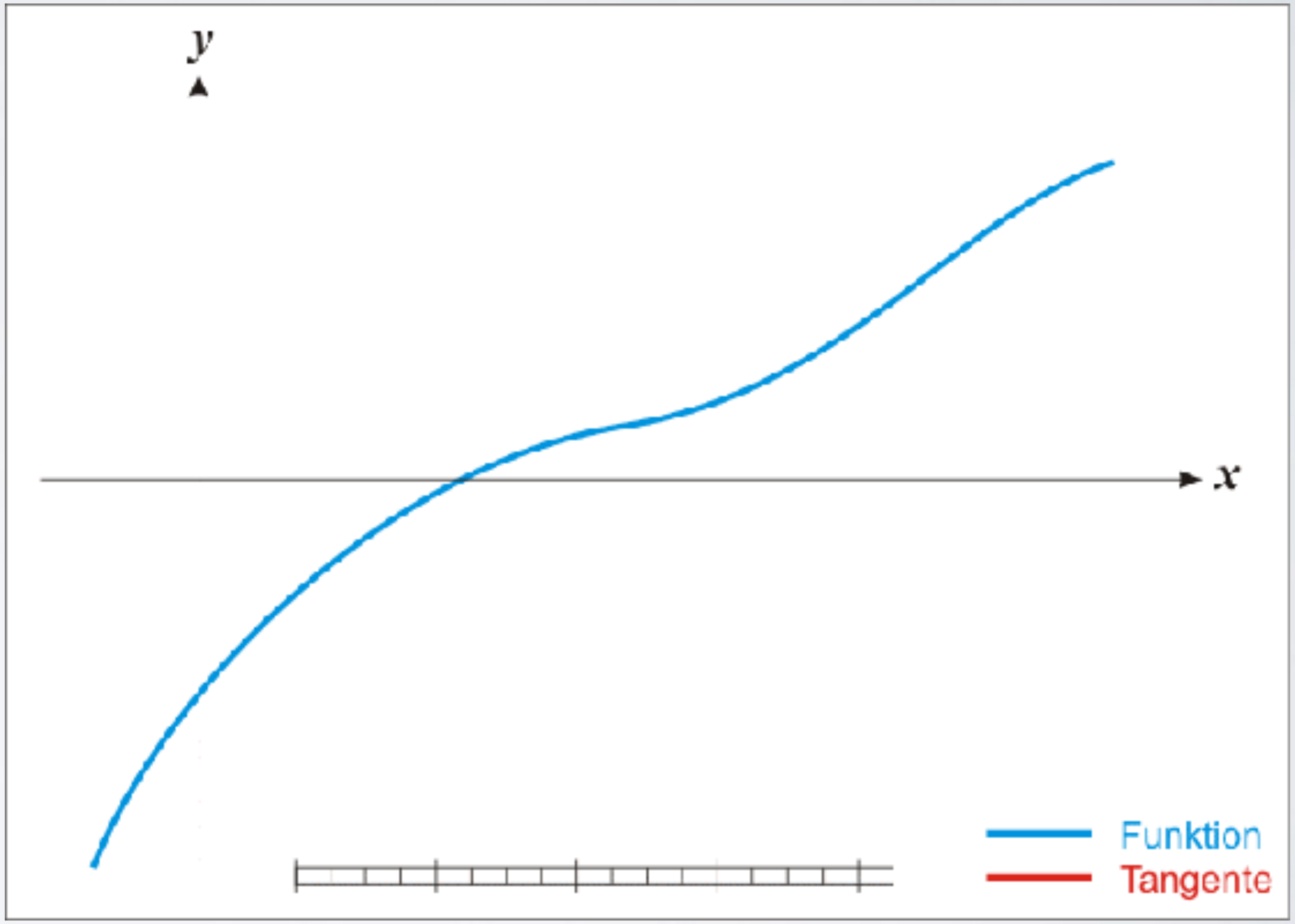
- Normalisation $f_0 - 1 = 0$

- Here, the discrete algebraic system $\vec{F}(\vec{X})$ is formed as

$$F_k = \begin{cases} (1-x_k^2)\left(\hat{D}^2\vec{X}\right)_k - 2x_k\left(\hat{D}\vec{X}\right)_k + X_{N+1}X_k & 0 \le k \le N \\ X_0 - 1 & k = N+1 \end{cases}$$

SOLUTION ALGORITHM

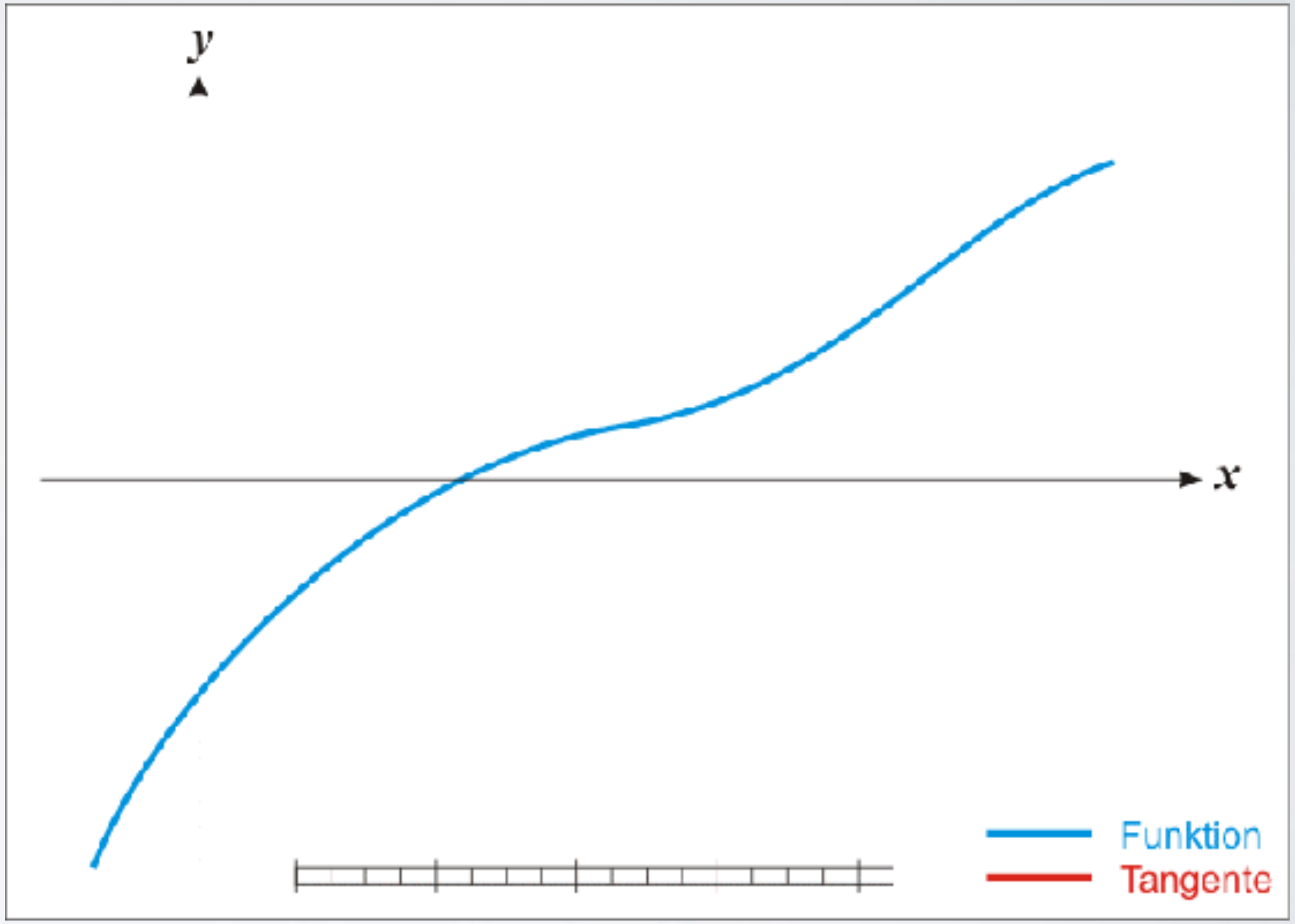
- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson method:



SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson method:

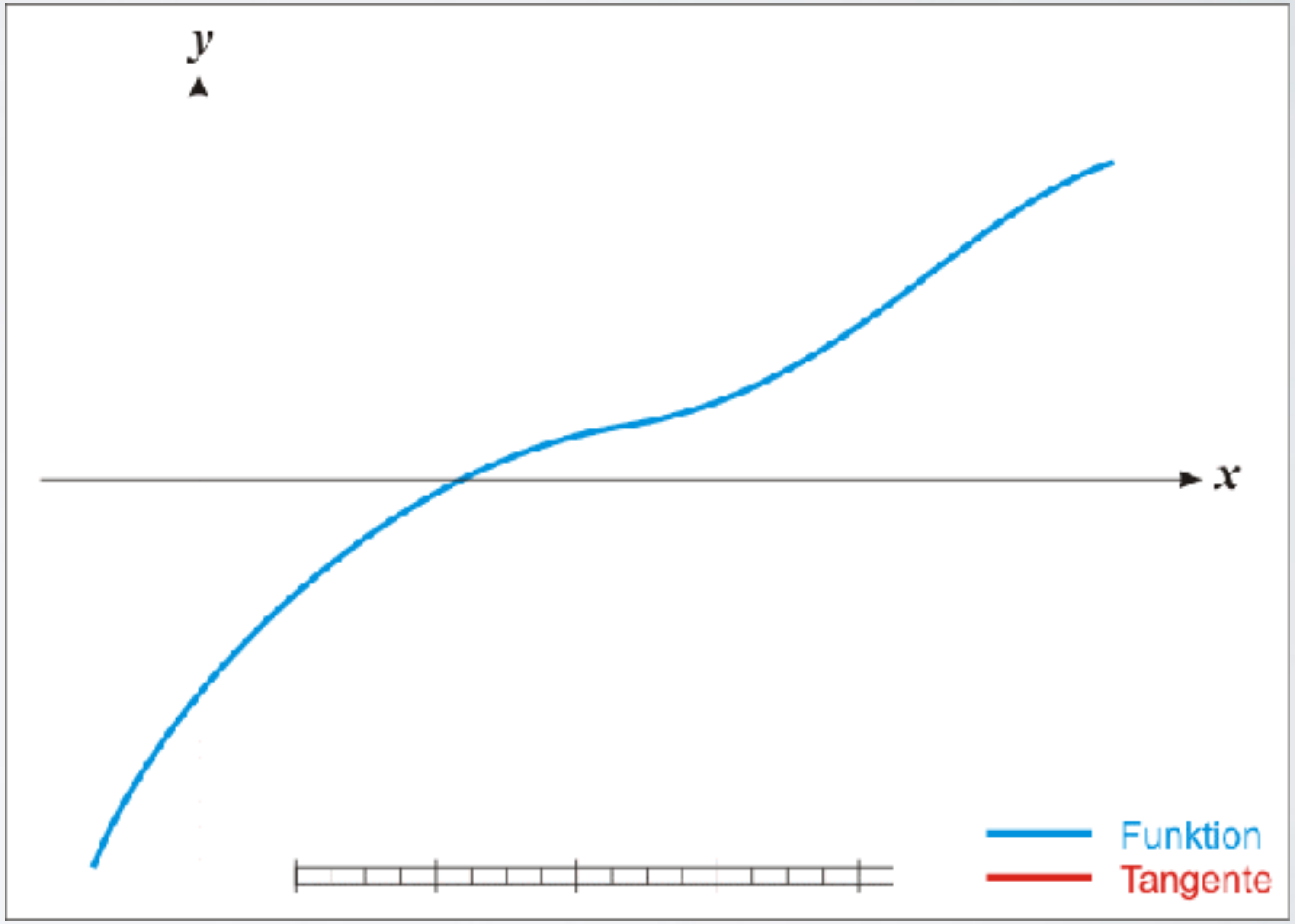
- **Newton method:** find roots $x^{(\mathrm{sol})}$ a function $F(x)$



SOLUTION ALGORITHM

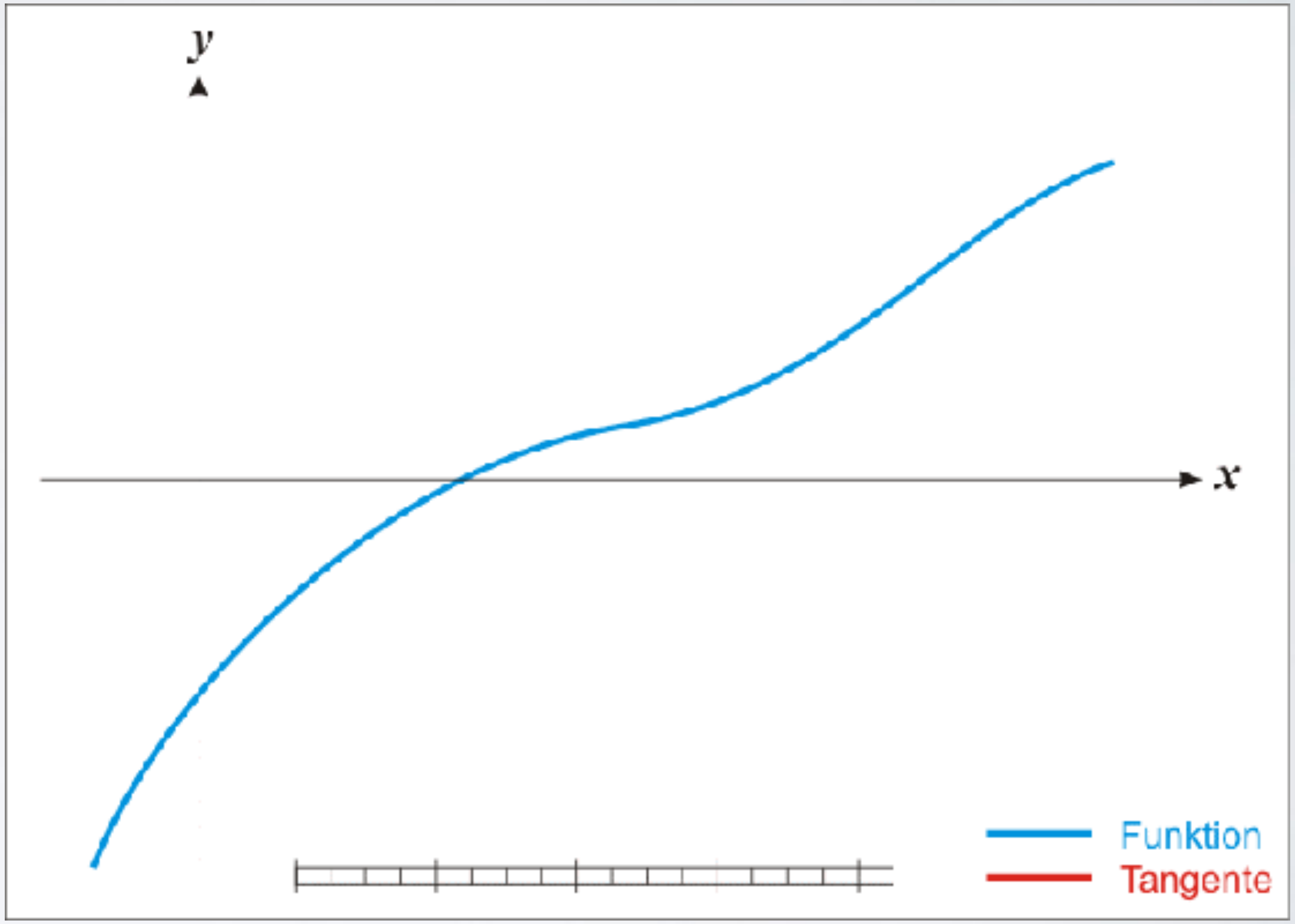
- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson method:

- **Newton method:** find solution $x^{(\mathrm{sol})}$ to the equation $F(x) = 0$



SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson method:
- **Newton method:** find solution $x^{(\mathrm{sol})}$ to the equation $F(x) = 0$

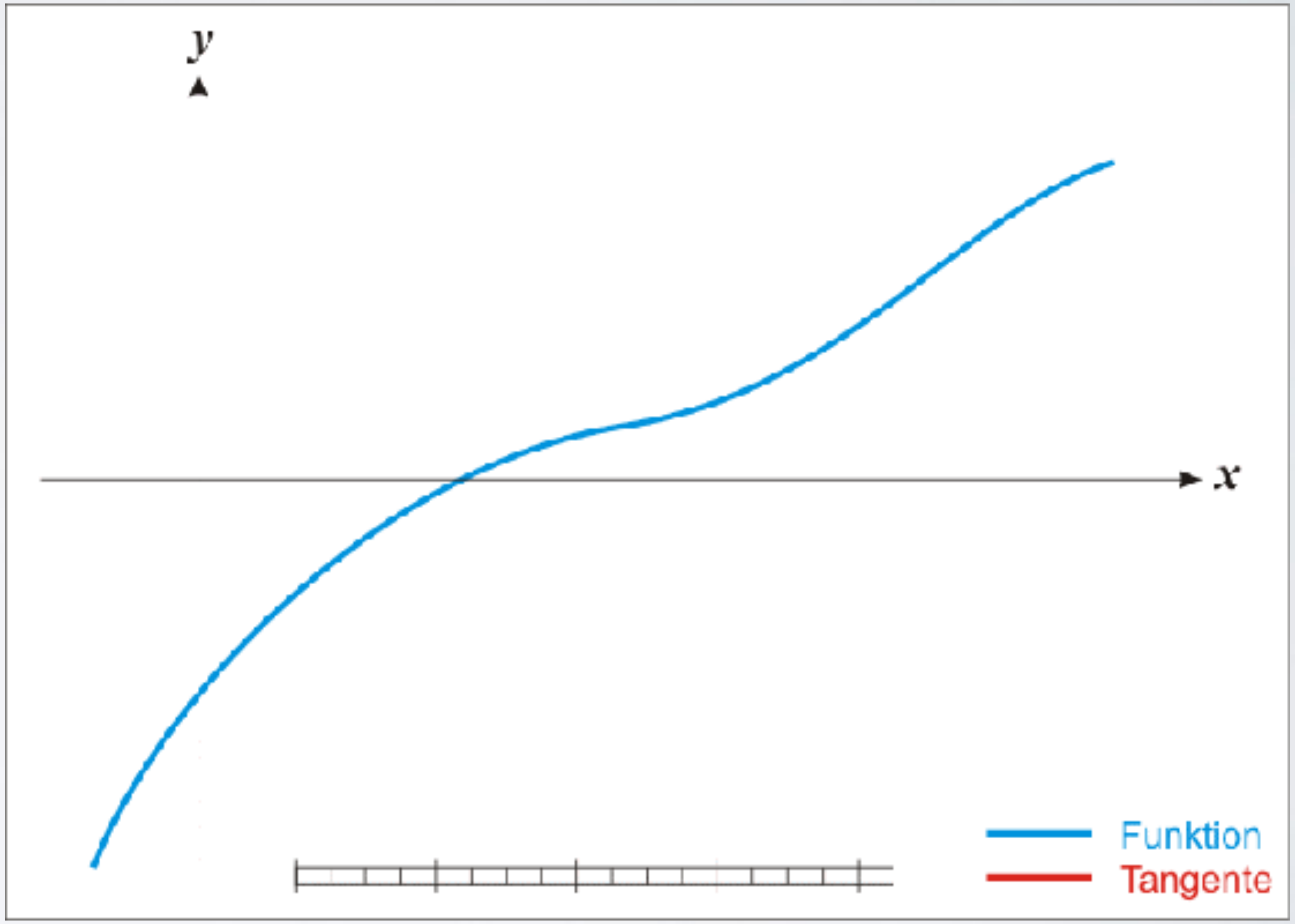


SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson method:

- **Newton method:** find solution $x^{(\mathrm{sol})}$ to the equation $F(x) = 0$

1. Give initial guess $x^{(0)}$

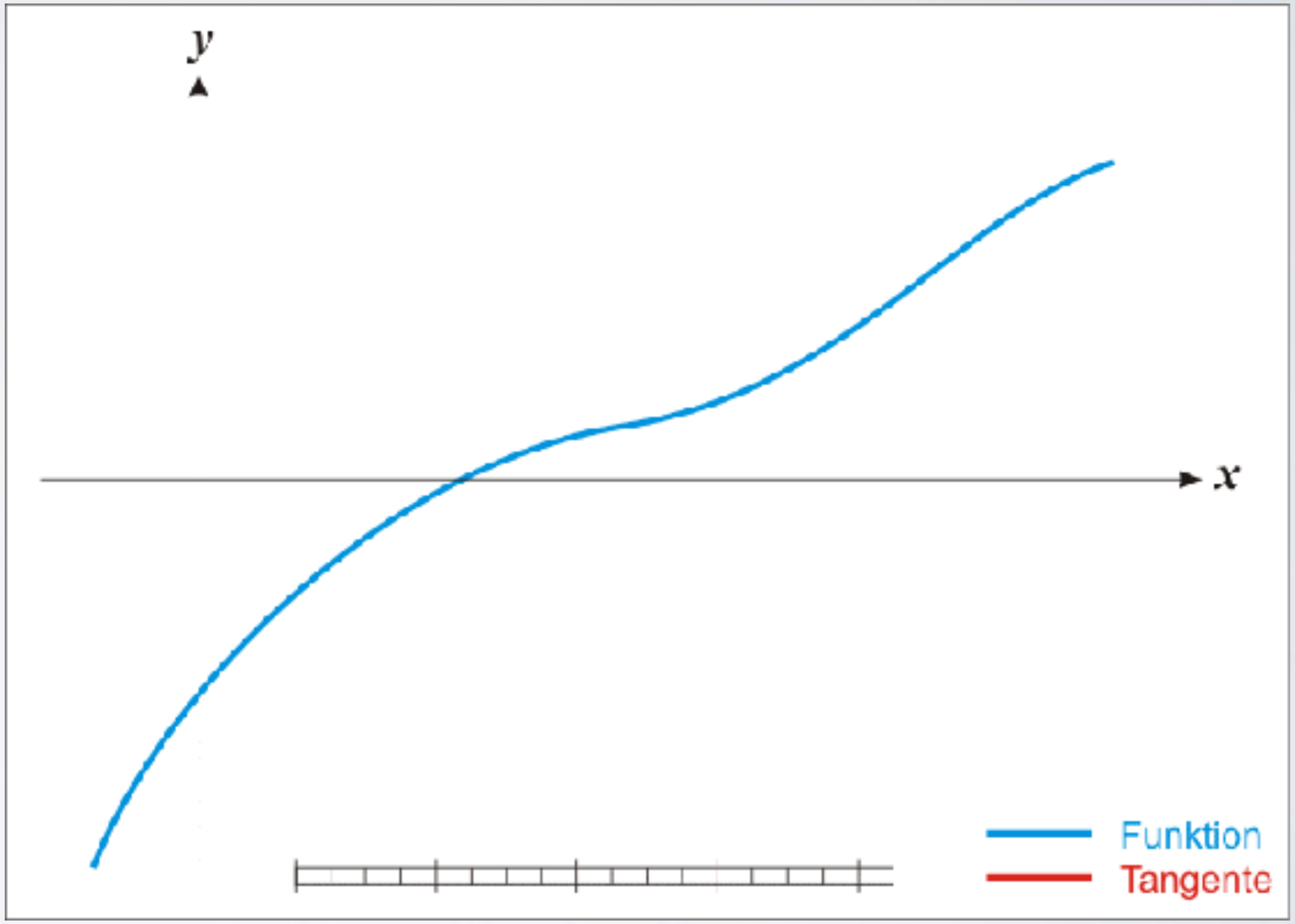


SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson method:

- **Newton method:** find solution $x^{(\mathrm{sol})}$ to the equation $F(x) = 0$

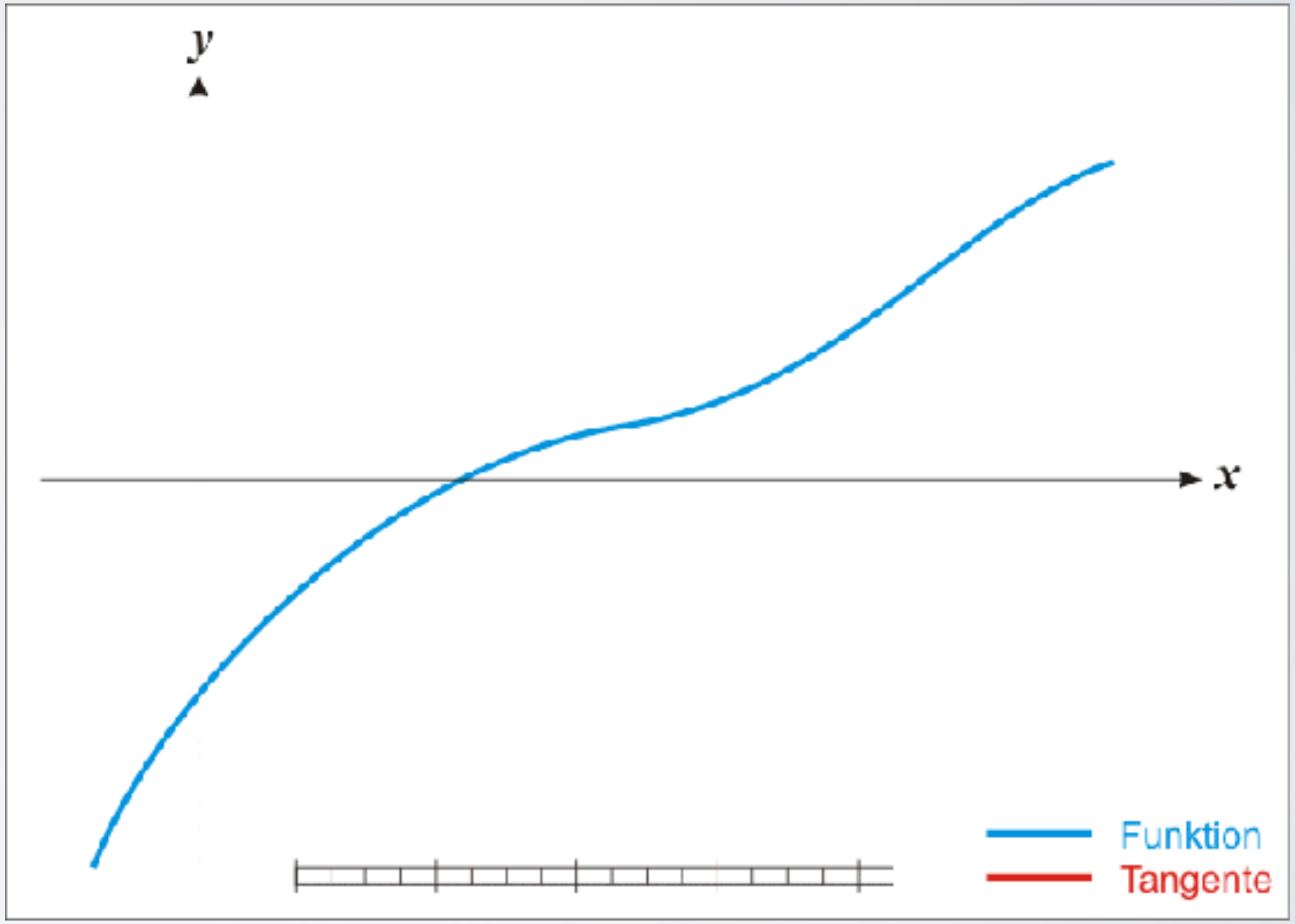
1. Give initial guess $x^{(0)}$
2. Calculate $f(x^{(0)})$



SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson method:
- **Newton method:** find solution $x^{(\mathrm{sol})}$ to the equation $F(x) = 0$

1. Give initial guess $x^{(0)}$
2. Calculate $f(x^{(0)})$
3. Calculate derivative $f'(x^{(0)})$

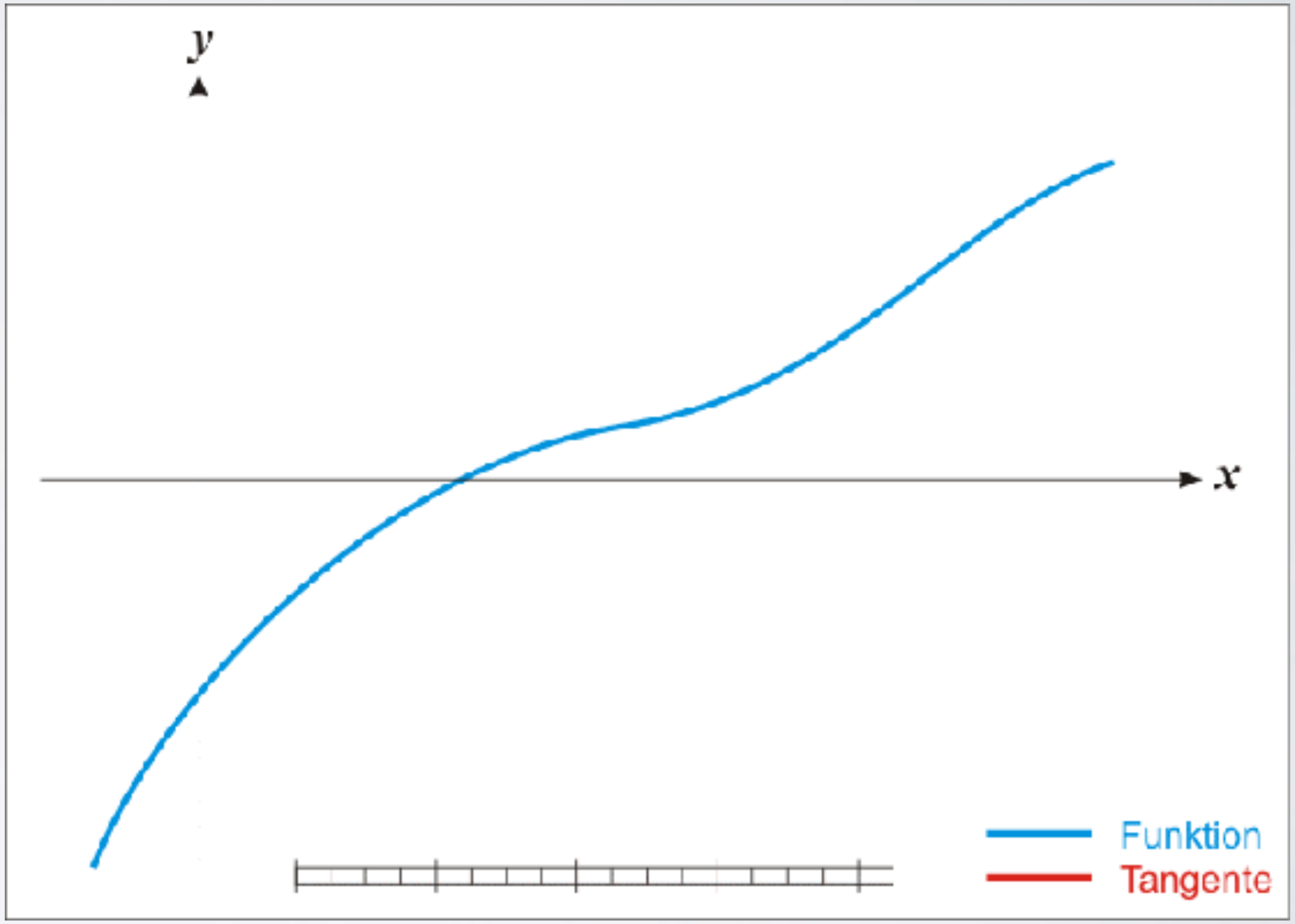


SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson method:
- **Newton method:** find solution $x^{(\mathrm{sol})}$ to the equation $F(x) = 0$

1. Give initial guess $x^{(0)}$
2. Calculate $f(x^{(0)})$
3. Calculate derivative $f'(x^{(0)})$
4. Identify $x^{(1)}$ via

$$x^{(1)} = x^{(0)} - \left[f'(x^0)\right]^{-1} f(x^{(0)})$$



SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson method:

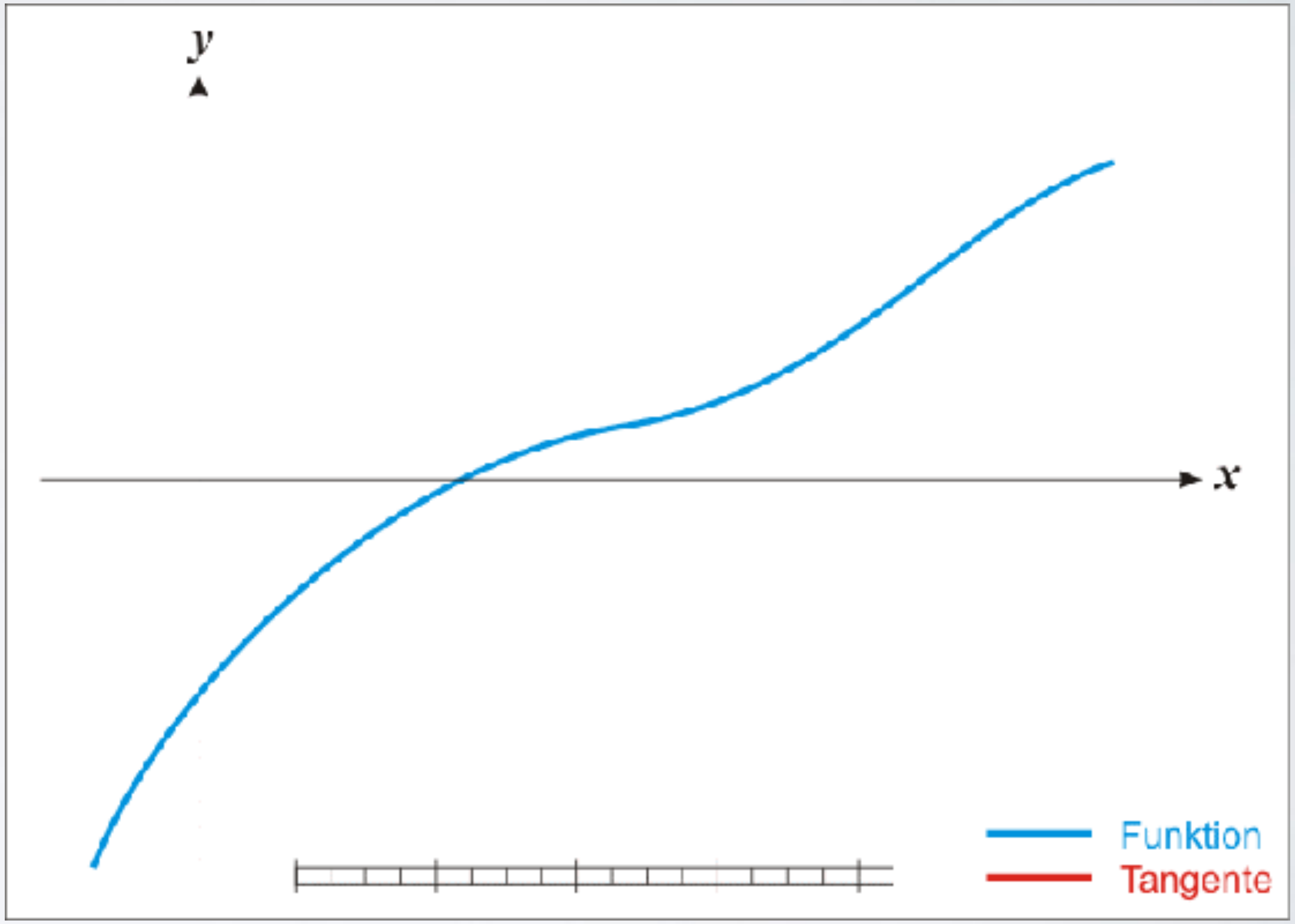
- **Newton method:** find solution $x^{(\mathrm{sol})}$ to the equation $F(x) = 0$

1. Give initial guess $x^{(0)}$
2. Calculate $f(x^{(0)})$
3. Calculate derivative $f'(x^{(0)})$
4. Identify $x^{(1)}$ via

$$x^{(1)} = x^{(0)} - \left[f'(x^0)\right]^{-1} f(x^{(0)})$$

5. Start again until

$$f(x^{(\mathrm{sol})}) \approx 0$$



SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson scheme:

SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson scheme:
  - Given initial-guess $\vec{X}^{(0)}$ (in general $\vec{F}(\vec{X}^{0}) \neq 0$)

SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson scheme:
  - Given initial-guess $\vec{X}^{(0)}$ (in general $\vec{F}(\vec{X}^0) \neq 0$)
  - Iterate to find a solution via

$$\vec{X}^{(m+1)} = \vec{X}^{(m)} - \left[\hat{J}\left(\vec{X}^{(m)}\right)\right]^{-1} \cdot \vec{F}(\vec{X}^{(m)})$$

where the Jacobian matrix is given by

$$J_{ij} = \frac{\partial F_i}{\partial X_j}$$

SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson scheme:
  - Given initial-guess $\vec{X}^{(0)}$ (in general $\vec{F}(\vec{X}^0) \neq 0$)
  - Iterate to find a solution via

$$\vec{X}^{(m+1)} = \vec{X}^{(m)} - \left[\hat{J}\left(\vec{X}^{(m)}\right)\right]^{-1} \cdot \vec{F}(\vec{X}^{(m)})$$

where the Jacobian matrix is given by

$$J_{ij} = \frac{\partial F_i}{\partial X_j}$$

- Solution is determined by $\vec{F}(\vec{X}^{(\mathrm{sol})}) < \mathrm{Tolerance}$

SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson scheme:
  - Given initial-guess $\vec{X}^{(0)}$ (in general $\vec{F}(\vec{X}^0) \neq 0$)
  - Iterate to find a solution via

$$\vec{X}^{(m+1)} = \vec{X}^{(m)} - \left[\hat{J}\left(\vec{X}^{(m)}\right)\right]^{-1} \cdot \vec{F}(\vec{X}^{(m)})$$

  where the Jacobian matrix is given by

$$J_{ij} = \frac{\partial F_i}{\partial X_j}$$

  - Solution is determined by $\vec{F}(\vec{X}^{(\mathrm{sol})}) < \mathrm{Tolerance}$
- Note that for the convergence of the scheme, a "good" initial guess $\vec{X}^{(0)}$ is necessary

SOLUTION ALGORITHM

- We find a solution $\vec{X}^{(\mathrm{sol})}$ using the Newton-Raphson scheme:
  - Given initial-guess $\vec{X}^{(0)}$ (in general $\vec{F}(\vec{X}^{0}) \neq 0$)
  - Iterate to find a solution via

$$\vec{X}^{(m+1)} = \vec{X}^{(m)} - \left[\hat{J}\left(\vec{X}^{(m)}\right)\right]^{-1} \cdot \vec{F}(\vec{X}^{(m)})$$

where the Jacobian matrix is given by

$$J_{ij} = \frac{\partial F_i}{\partial X_j}$$

- Solution is determined by $\vec{F}(\vec{X}^{(\mathrm{sol})}) < \mathrm{Tolerance}$

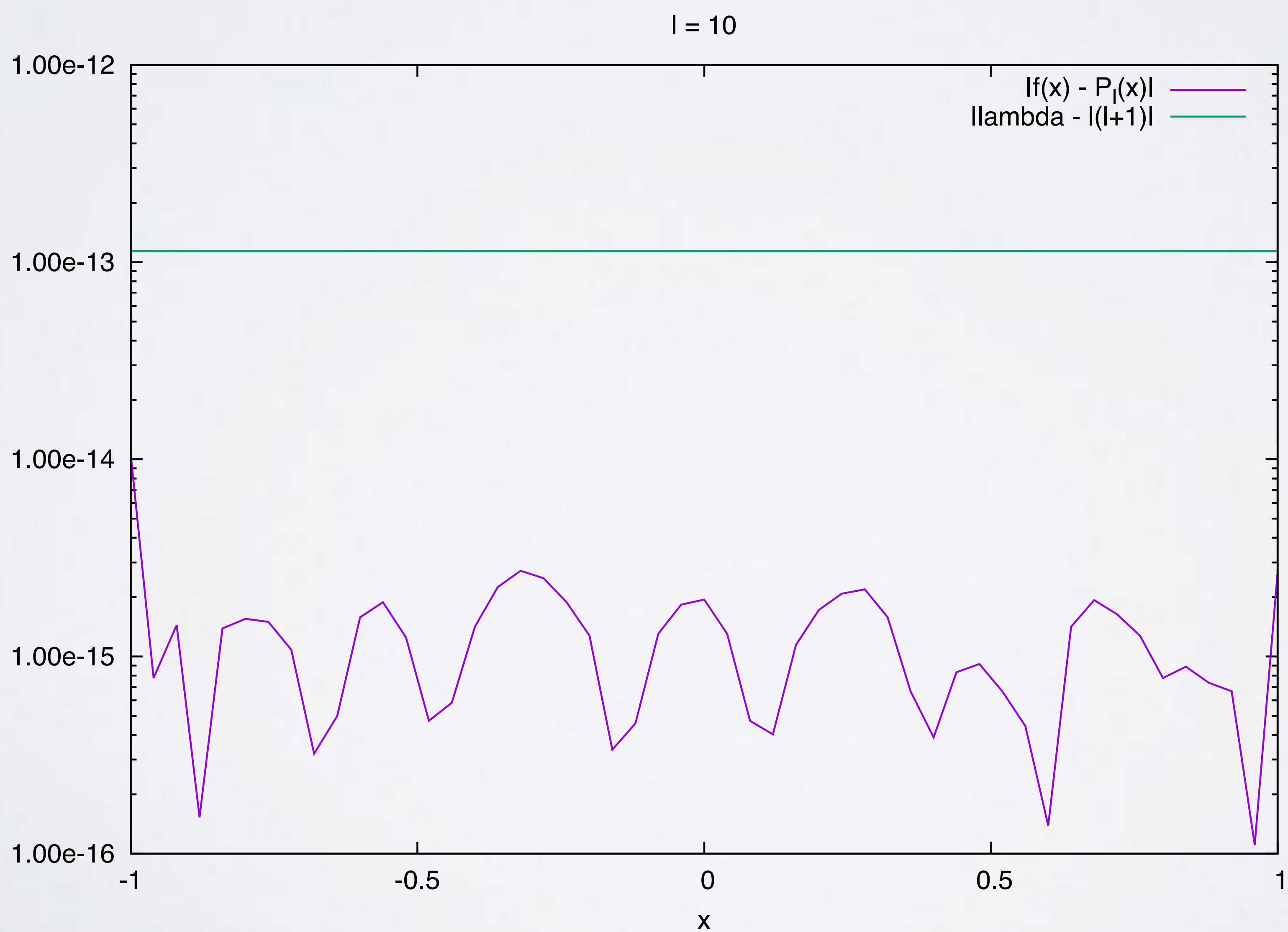
- Note that for the convergence of the scheme, a "good" initial guess $\vec{X}^{(0)}$ is necessary
- In higher dimensions, inverting the Jacobian matrix is expensive and we need to make use of further iterative methods to speed up the algorithm

ODE (EIGENVALUE PROBLEM)
SOLUTION

- Input: parameter ℓ to indicate the order of the eigenvalue
- Input: initial guess for the Newton-Raphson Scheme

$$f^{(0)}(x) = x^{\ell} - x^{\ell-2} + x^{\ell-4} - \cdots \qquad \lambda^{(0)} = \ell^2$$

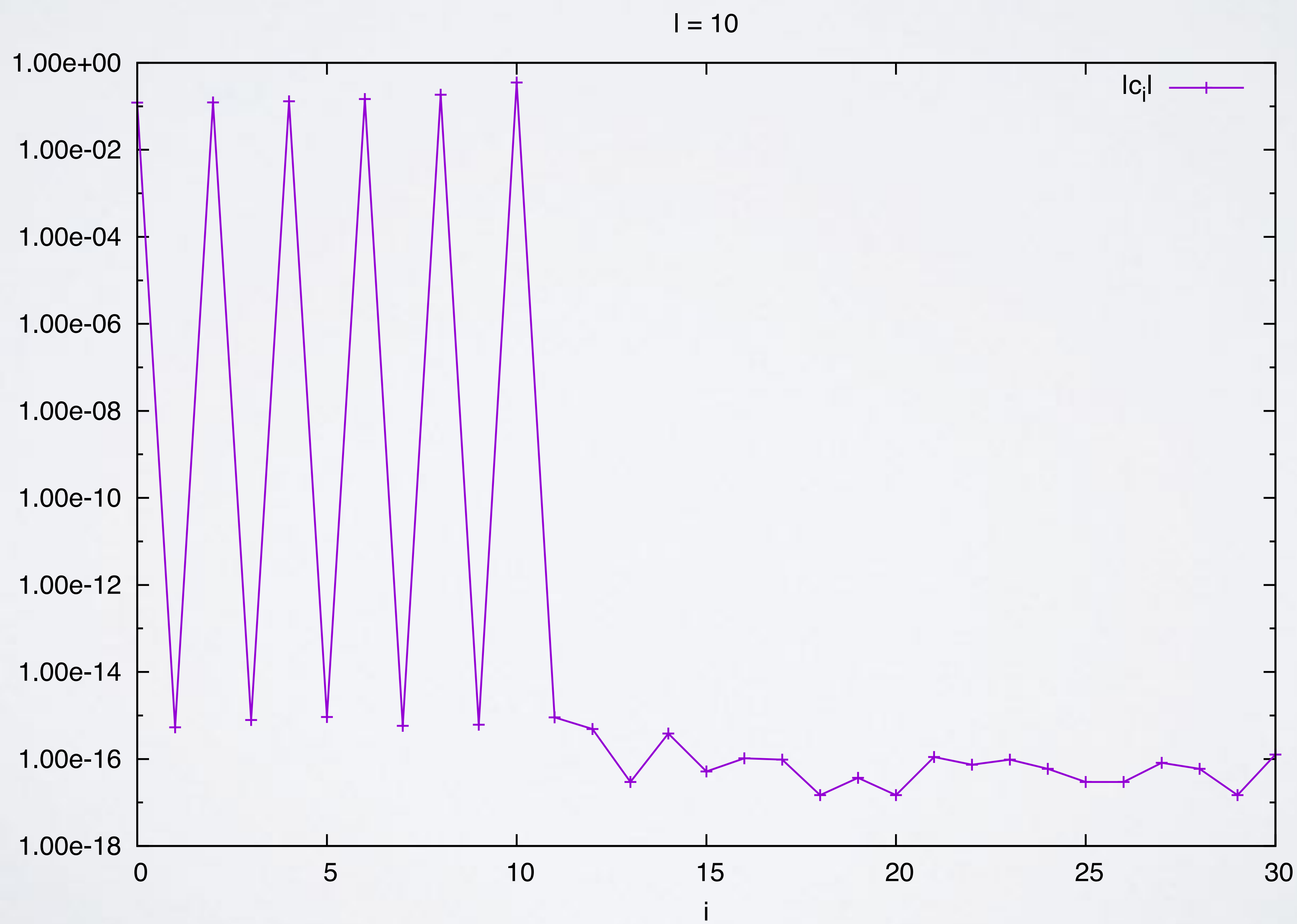
Error



ODE (EIGENVALUE PROBLEM)
SOLUTION

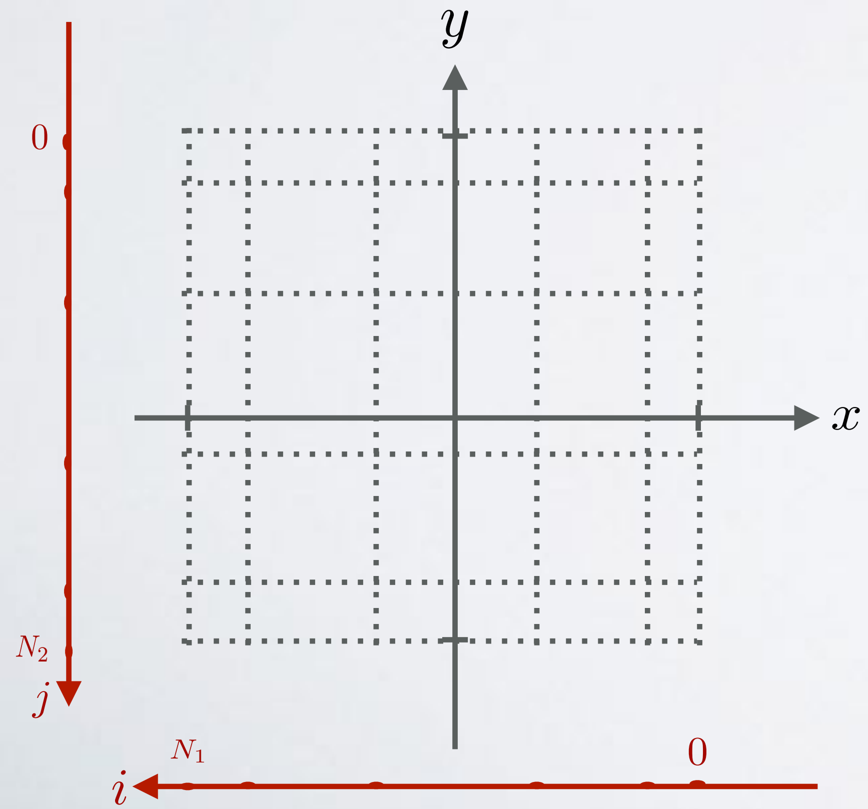
- Input: parameter ℓ to indicate the order of the eigenvalue
- Input: initial guess for the Newton-Raphson Scheme

$$f^{(0)}(x) = x^{\ell} - x^{\ell-2} + x^{\ell-4} - \cdots \qquad \lambda^{(0)} = \ell^2$$



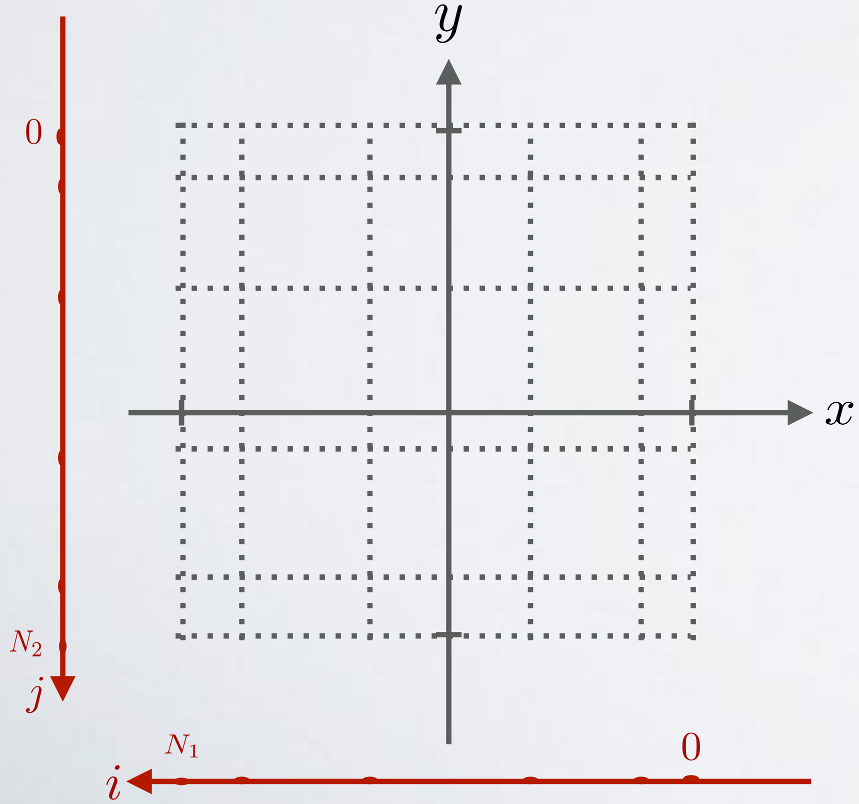
HIGHER DIMENSION

- **Extend the same idea to higher dimension:**



HIGHER DIMENSION

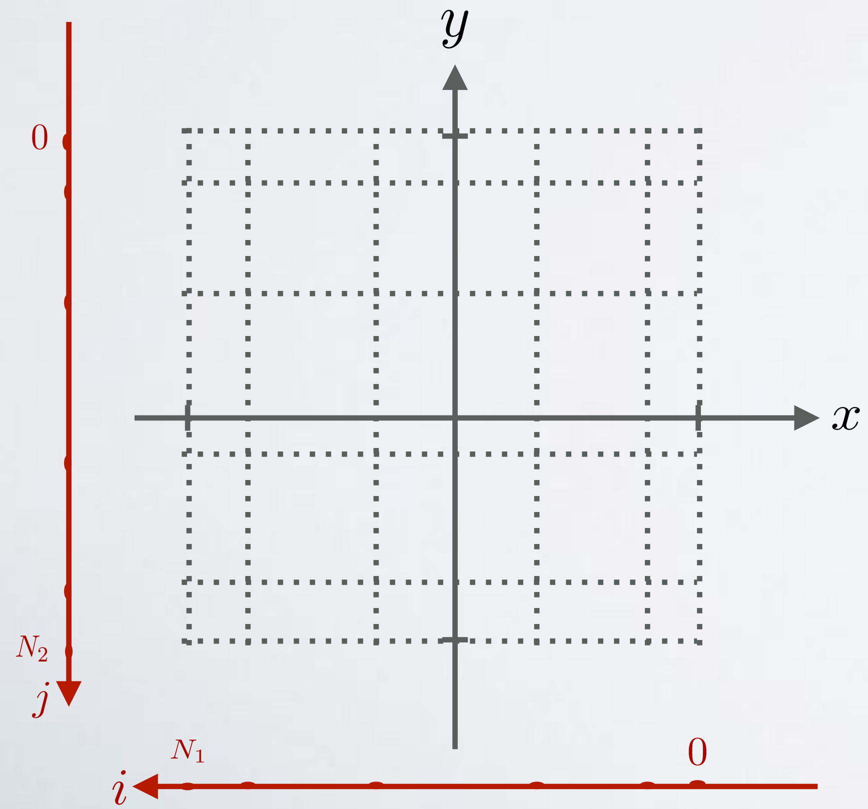
- **Extend the same idea to higher dimension:**
  - Vector $\vec{X}$ stores all unknown of the system



$$\vec{X} = \begin{pmatrix} f(x_0, y_0) \\ \vdots \\ f(x_0, y_{N2}) \\ \vdots \\ f(x_{N_1}, y_0) \\ \vdots \\ f(x_{N_1}, y_{N_2}) \\ -- \\ \text{extra} \quad \text{unknown} \end{pmatrix}$$

HIGHER DIMENSION

- **Extend the same idea to higher dimension:**
  - Vector $\vec{X}$ stores all unknown of the system
  - Algebraic system $\vec{F}(\vec{X})$ is composed by the partial differential equations, boundary conditions and subsidiary conditions



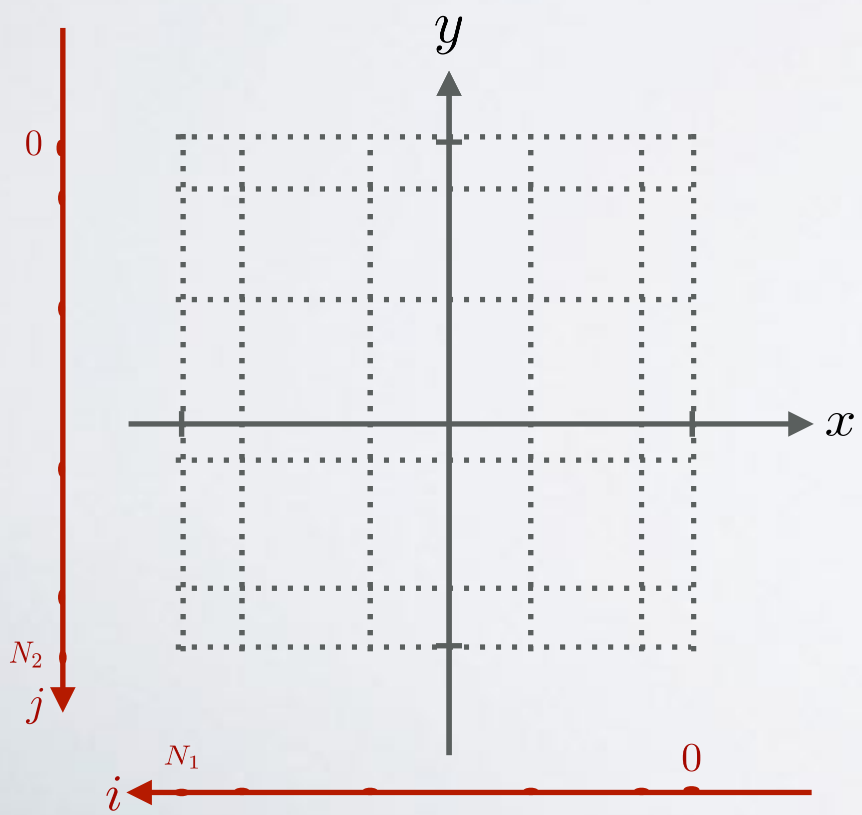
$$\vec{X} = \begin{pmatrix} f(x_0, y_0) \\ \vdots \\ f(x_0, y_{N2}) \\ \vdots \\ f(x_{N_1}, y_0) \\ \vdots \\ f(x_{N_1}, y_{N_2}) \\ -- \\ \text{extra} \quad \text{unknown} \end{pmatrix}$$

$$F_k(\vec{X}) = \begin{cases} \mathrm{BC}_{x_0}(y_j) & i = 0 & 0 \le j \le N_2 \\ \mathrm{BC}_{x_{N_1}}(y_j) & i = N_1 & 0 \le j \le N_2 \\ \mathrm{BC}_{y_0}(x_i) & 1 \le i \le N_1 - 1 & j = 0 \\ \mathrm{BC}_{y_{N_2}}(x_i) & 1 \le i \le N_1 - 1 & j = N_2 \\ \mathrm{PDE}(x_i, y_k) & 1 \le i \le N_1 - 1 & 1 \le j \le N_2 - 1 \\ \text{subsidiary} \quad \text{equations} & k = \text{extra} \quad \text{components} & \end{cases}$$

HIGHER DIMENSION

- **Extend the same idea to higher dimension:**
  - Vector $\vec{X}$ stores all unknown of the system
  - Algebraic system $\vec{F}(\vec{X})$ is composed by the partial differential equations, boundary conditions and subsidiary conditions
  - Solution is found via Newton-Raphson scheme (eventually with further tricks to speed up the inversion of the Jacobian Matrix)

$$\vec{X} = \begin{pmatrix} f(x_0, y_0) \\ \vdots \\ f(x_0, y_{N2}) \\ \vdots \\ f(x_{N_1}, y_0) \\ \vdots \\ f(x_{N_1}, y_{N_2}) \\ -- \\ \text{extra} \quad \text{unknown} \end{pmatrix}$$



$$F_k(\vec{X}) = \begin{cases} \mathrm{BC}_{x_0}(y_j) & i = 0 & 0 \le j \le N_2 \\ \mathrm{BC}_{x_{N_1}}(y_j) & i = N_1 & 0 \le j \le N_2 \\ \mathrm{BC}_{y_0}(x_i) & 1 \le i \le N_1 - 1 & j = 0 \\ \mathrm{BC}_{y_{N_2}}(x_i) & 1 \le i \le N_1 - 1 & j = N_2 \\ \mathrm{PDE}(x_i, y_k) & 1 \le i \le N_1 - 1 & 1 \le j \le N_2 - 1 \\ \text{subsidiary} \quad \text{equations} & k = \text{extra} \quad \text{components} & \end{cases}$$