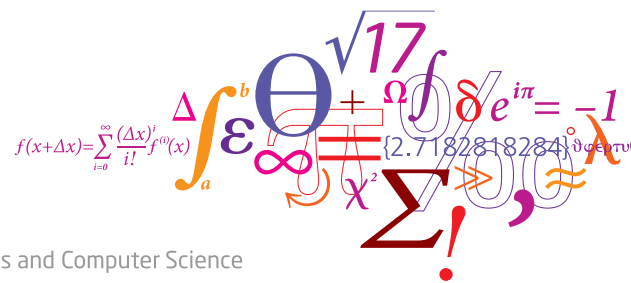
Lecture 5 Boundary Value Problems

02689 Advanced numerical methods for differential equations

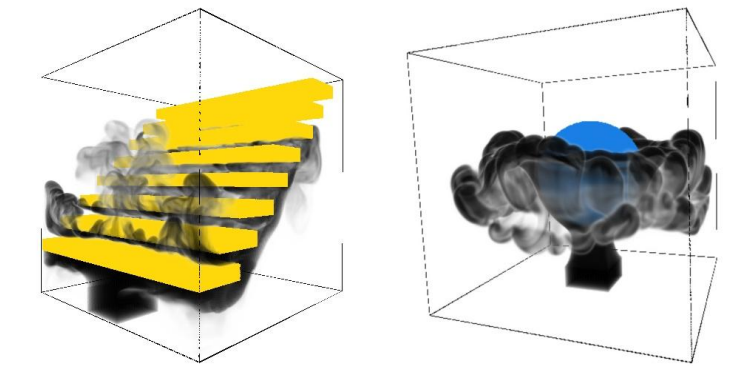
Allan P. Engsig-Karup
Scientific Computing Section
DTU Compute
Technical University of Denmark



DTU Compute
Department of Applied Mathematics and Computer Science

Examples of Boundary Value Problems

Smoke simulation based on computational fluid dynamics that rely on a Poisson solver...



Source: Glimberg, Erleben, Bennetsen (2009)

Solution of Boundary Value Problems

We are concerned with two choices for the construction of numerical methods for the solution of differential equations

- How to represent the solution?
- How to satisfy the differential equations?

Goals

- Generate system of algebraic equations to solve for unknowns of our spectral numerical scheme.

For the solution of differential equations using Spectral Methods we will focus on

- Choice of function approximations.
- Method of Weighted Residual Methods (MWR) for satisfying the differential equations and boundary conditions.

Measuring errors

In Spectral Methods we construct approximations to $u(x)$

$$u_N(x) = \sum_{n=0}^{N} \hat{u}_n \phi_n(x)$$

Introduce the residual as a measure of error for

- **Function approximation** of $u(x)$

  $$\mathcal{R}_N(x) = u(x) - u_N(x)$$

  Measures the accuracy of the approximate solution.

- **Approximate solution** of a differential equation

  $$\mathcal{L}u(x) - f(x) = 0$$

  where for some approximate solution

  $$\mathcal{R}_N(x) = \mathcal{L}u_N(x) - f(x)$$

  Measures how well the differential equation is satisfied.

Goal is to make some $||\mathcal{R}_N||$ of the residuals sufficiently small.

Method of Weighted Residuals (MWR)

The Method of Weighted Residuals (MWR) was coined by Crandall (1956) and described in a first review by Finnlayson and Scriven (1966) as

> *The Method of Weighted Residuals unifies many approximate methods of solution of differential equations that are being used currently.*

and

> *The Method of Weighted Residuals is an engineer's tool for finding approximate solutions to the equations of change of distributed systems.*

In short, the MWR method unifies in a systematic way several common discretization methods.

Method of Weighted Residuals (MWR)

In general, require the residual $\mathcal{R}_N(x)$ to vanish in the sense

$$(\mathcal{R}_N, \psi_i)_w = \int_a^b \mathcal{R}_N(x)\psi_i(x)w(x)dx = 0, \quad i = 0, 1, ..., N$$

where $\psi_i(x)$ is referred to as trial or test functions and $w(x) > 0$ weight function for $x \in [a, b]$.

The choice of test and weight functions defines the discretization method. For example

Method	**Test function**
Galerkin (Tau)	$\psi_i(x) = \phi_i(x)$
Collocation	$\psi_i(x) = \delta(x - x_i)$
Petrov-Galerkin	$\psi_i(x) \neq \phi_i(x)$

where $\phi_i(x)$, $i = 0, 1, ..., N$ is the set of basis or trial functions used in the function approximation of $u(x)$.

Function approximation by MWR

The MWR generalizes the way we determine coefficients $\hat{\mathbf{u}}_k$ in series expansions of $u(x)$.

Assume, basis functions $\{\phi_k\}_{k=0}^N$ are orthogonal with weight $w(x)$

$$(\phi_i, \phi_j)_w = \gamma_i \delta_{ij}$$

and define the residual of the function approximation as

$$\mathcal{R}_N(x) = u(x) - u_N(x) = u(x) - \sum_{n=0}^{N} \hat{u}_n \phi_n(x)$$

Function approximation by MWR

Then, if we require the residual to vanish in the mean sense by a Galerkin method

$$(\mathcal{R}_N, \phi_i)_w = \int_a^b \left(u - \sum_{n=0}^{N} \hat{u}_n \phi_n \right) \phi_i w dx = 0, \quad i = 0, 1, ..., N$$

We find by orthogonality of the basis and test functions that the coefficients can be evaluated as

$$\hat{u}_n = \frac{(u, \phi_n)_w}{(\phi_n, \phi_n)_w}, \quad n = 0, 1, ..., N$$

Which is recognized as the continuous (exact) expression for expansion coefficients.

Function approximation by MWR

Then, if we require the residual to vanish in the point-wise sense by a Collocation method ($\psi_i = \delta_i$ and $w(x) = 1$)

$$\sum_{n=0}^{N} \tilde{u}_n \phi_n(x_i) = u(x_i), \quad i = 0, 1, ..., N$$

we find an algebraic system of equations for determining the $(N+1)$ coefficients $\hat{\mathbf{u}}$ (can be modal/nodal).

Remark: These residual conditions corresponds to the conditions of the interpolating polynomial based on the set of points $\{x_i\}_{i=0}^{N}$.

Explicit expressions for determining the expansion coefficients can be found by exploiting discrete orthogonality of special basis functions and special choices of collocation points

$$\tilde{u}_n = \frac{(u, \phi_n)_{w,N}}{(\phi_n, \phi_n)_{w,N}}, \quad n = 0, 1, ..., N$$

Solution of differential equations by MWR

How can we solve Boundary Value Problems with Spectral Methods?

Consider in the following general linear second-order differential equation with inhomogenous boundary conditions (e.g. Dirichlet, Neumann and Robin) for $u(x)$

$$\begin{aligned} \mathcal{L}u - f &= 0, \quad \alpha < x < \beta \\ \mathcal{B}_- u &= g_-, \quad x = \alpha \\ \mathcal{B}_+ u &= g_+, \quad x = \beta \end{aligned}$$

where $\mathcal{L}$, $\mathcal{B}_-$ and $\mathcal{B}_+$ are differential operators.

Solution of differential equations by MWR

The problem can be reformulated to an equivalent homogeneous formulation by substitution of

$$u(x) = u_0(x) + v(x)$$

where $u_0(x)$ is an arbitrary function satisfying the boundary conditions. The new problem for $v(x)$ is

$$\begin{aligned} \mathcal{L}v - h &= 0, \quad \alpha < x < \beta \\ \mathcal{B}_- v &= 0, \quad x = \alpha \\ \mathcal{B}_+ v &= 0, \quad x = \beta \end{aligned}$$

where $h = f - \mathcal{L}u_0$.

Solution of differential equations by MWR

In the Galerkin method the basis functions $\{\phi_n\}_{n=0}^N$ are by choice required to satisfy the homogeneous boundary conditions of the BVP

$$\begin{aligned} \mathcal{B}_-\phi_n &= 0, \quad x = \alpha \\ \mathcal{B}_+\phi_n &= 0, \quad x = \beta \end{aligned}$$

In this case, the solution $u(x)$ is sought in the combination

$$u(x) = u_0(x) + v(x)$$

where $u_0(x)$ is selected to satisfy the BCs.

Therefore, we seek to determine an approximate solution $v_N(x)$ to the homogenous BVP in the form

$$v_N(x) = \sum_{n=0}^{N} \hat{v}_n \phi_n(x)$$

Solution of differential equations by MWR

Define the residual

$$\mathcal{R}_N(x) = \mathcal{L}v_N - h$$

Require the residual to vanish in the mean sense in a weak (integral) formulation by MWR

$$(\mathcal{R}_N, \phi_i)_w = (\mathcal{L}v_N - h, \phi_i)_w, \quad i = 0, 1, ..., N$$

Insert expression for v_N and we find the algebraic equations

$$\boxed{\sum_{n=0}^{N} \hat{v}_n(\mathcal{L}\phi_n, \phi_i)_w = (h, \phi_i)_w = \hat{h}_i, \quad i = 0, 1, ..., N}$$

where the weighted inner product can be accurately evaluated using properties of the basis functions.

This makes it possible to generate a linear system of equations

$$\mathcal{L}_N \hat{\mathbf{u}} = \hat{\mathbf{h}}, \quad \mathcal{L}_N \in \mathbb{R}^{(N+1)\times(N+1)}, \quad \hat{\mathbf{u}}, \hat{\mathbf{h}} \in \mathbb{R}^{(N+1)}$$

with modal solution $\hat{\mathbf{u}} = \mathcal{L}_N^{-1}\hat{\mathbf{h}}$.

Solution of differential equations by MWR

Not everything is periodic and it is in general not straightforward to select orthogonal basis functions that satisfies the homogenous boundary conditions.

The Tau Method is a generalization of the Galerkin Method and does not require the basis functions $\{\phi_n\}_{n=0}^N$ to be chosen to satisfy the homogeneous boundary conditions.

In the Tau Method the solution $u(x)$ is sought in the form

$$u_N(x) = \sum_{n=0}^{N} \hat{u}_n \phi_n(x), \quad \mathcal{R}_N(x) = \mathcal{L}u_N - f$$

As in the Galerkin Method, the residual is required to vanish in the mean sense

$$(\mathcal{R}_N, \phi_i)_w = (\mathcal{L}u_N - f, \phi_i)_w, \quad i = 0, 1, ..., N$$

Solution of differential equations by MWR

Insert expression for u_N and we find the $(N-1)$ algebraic equations

$$\boxed{\sum_{n=0}^{N} \hat{u}_n(\mathcal{L}\phi_n, \phi_i)_w = (f, \phi_i)_w = \hat{f}_i, \quad i = 0, 1, ..., N-1}$$

Two more conditions are needed and is found by requiring that the expansion coefficients satisfies the two boundary conditions

$$\boxed{\mathcal{B}_- u_N(\alpha) = g_-, \quad \mathcal{B}_+ u_N(\beta) = g_+}$$

This makes it possible to generate a linear system of equations

$$\mathcal{L}_N \hat{\mathbf{u}} = \hat{\mathbf{f}}, \quad \mathcal{L}_N \in \mathbb{R}^{(N+1)\times(N+1)}, \quad \hat{\mathbf{u}}, \hat{\mathbf{f}} \in \mathbb{R}^{(N+1)}$$

with modal solution $\hat{\mathbf{u}} = \mathcal{L}_N^{-1}\hat{\mathbf{f}}$.

Solution of differential equations by MWR

In the Collocation Method the solution is sought in one of

$$u_N(x) = \sum_{n=0}^{N} \hat{u}_n \phi_n(x) = \sum_{n=0} u_n h_n(x), \quad \mathcal{R}_N(x) = \mathcal{L} u_N - f$$

The residual is required to vanish in a point-wise sense at inner collocation points

$$\boxed{\mathcal{R}_N(x_i) = \mathcal{L}_N u_N(x_i) - f(x_i), \quad i = 0, 1, ..., N-2}$$

and at the two outer collocation points for the boundary nodes.

$$\boxed{\mathcal{B}_- u_N(x_0) = g_-, \quad \mathcal{B}_+ u_N(x_N) = g_+}$$

This makes it possible to generate a linear system of equations

$$\mathcal{L}_N \mathbf{u} = \mathbf{f}, \quad \mathcal{L}_N \in \mathbb{R}^{(N+1)\times(N+1)}, \quad \mathbf{u}, \mathbf{f} \in \mathbb{R}^{(N+1)}$$

with nodal solution $\mathbf{u} = \mathcal{L}_N^{-1} \mathbf{f}$. Modal solution also possible.

Method of Weighted Residuals (MWR)

In summary,

- **Galerkin**. Select or construct a set of basis functions which all satisfy the boundary conditions and require residual to be orthogonal to as many of the basis functions as possible. Test functions are chosen from the set of basis functions.
  Idea: Enforce BCs via convenient choice of basis functions.
- **Tau**. Require that expansion coefficients are selected to enforce boundary conditions and residual to be orthogonal to as many of the test functions as possible.
  Idea: Choose whatever basis functions is convenient and introduce constraints to enforce BCs.
- **Collocation**. Require that expansion coefficients are selected to enforce boundary conditions and require residual to be zero at as many spatial points as possible.
  Idea: Enforce both differential equations and constraints at every node. Choose basis functions conveniently.

To illustrate these ideas, let's consider some examples...

Strong and weak formulations

Differential equations can be formulated in different equivalent ways

- **Strong form** (classical differential form)
  The differential equation and boundary conditions are satisfied at each point in domain at each point in time.

$$-\frac{d}{dx}\left(a\frac{du}{dx}\right) + b\frac{du}{dx} + cu = f(x), \quad x \in [-1, 1]$$

- **Weak form** (integral form)
  A weighted integral of the differential equation is satisfied and boundary conditions can be incorporated.

$$\int_{-1}^{1} \left[-\frac{d}{dx}\left(a\frac{du}{dx}\right) + b\frac{du}{dx} + cu\right] v(x)w(x)dx = \int_{-1}^{1} f(x)v(x)w(x)dx$$

If sufficient smoothness of functions are assumed then the continuous formulations can be shown to be equivalent.

Remark, the weak formulation allows the possible benefit of domain decompositions.

Basic spectral discretization methods

In general, the solution of BVPs using a Spectral Method follows this recipe

- Formulate well-posed problem in strong or weak form.
- Discretize problem via formulation
  - Choose how to represent solution.
  - Choose how to satisfy differential equations (by MWR) and enforce boundary conditions.
- Generate algebraic systems of equations and solve!

Basic spectral discretization methods

To illustrate basic spectral discretization methods, we introduce the linear differential operator

$$\mathcal{L} = \frac{d}{dx}\left(a\frac{d}{dx}\right) + b\frac{d}{dx} + c$$

The strong form of this model problem is

$$\begin{aligned} \mathcal{L}u(x) &= f(x), \qquad x \in [\alpha, \beta] \\ u(-1) &= u(1) = 0 \quad (\text{hom. Dirichlet}) \end{aligned}$$

and the weak form starts from the weighted integral formulation

$$\int_\alpha^\beta \mathcal{L}u(x)v(x)w(x)dx = \int_\alpha^\beta f(x)v(x)w(x)dx, \quad \forall v(x)$$

with boundary conditions incorporated

- Implicitly via choice of basis functions.
- In formulation via boundary terms that results from integration by parts.
- Additional constraints or penalization.

Fourier Galerkin Method

Seek an approximation to $u(x) \in C_p^\infty([0, 2\pi])$ in the modal form

$$u_N(x) = \sum_{k=-N/2}^{N/2} \hat{u}_k e^{ikx}$$

The residual of the equation can then be formed

$$\mathcal{R}_N u(x) = \mathcal{L} u_N - f(x) = \sum_{k=-N/2}^{N/2} \hat{u}_k \mathcal{L} e^{ikx} - f(x)$$

Require the residual in the mean to vanish with test functions $\psi_n(x) = e^{inx}$, we find for $n = -N/2, ..., N/2$

$$(\mathcal{R}_N u(x), \psi_j(x)) = \sum_{k=-N/2}^{N/2} \hat{u}_k (\mathcal{L} e^{ikx}, e^{inx}) - (f(x), e^{inx}) = 0$$

Fourier Galerkin Method

By direct differentiation of the basis functions we find

$$\mathcal{L}e^{ikx} = g(k)e^{ikx}, \quad g(k) = (ak^2 + ikb + c)e^{ikx}$$

Therefore, the residual equations simplifies such that

$$\sum_{k=-N/2}^{N/2} \hat{u}_k(ak^2 + ikb + c)(e^{ikx}, e^{inx}) - (f(x), e^{inx}) = 0$$

The Fourier Galerkin Method is expressed in terms of algebraic equations for the $(N+1)$ modal coefficients as

$$\boxed{g(k)\hat{u}_n = \hat{f}_n, \quad n = -N/2, ..., N/2}$$

under assumption of exact integration for inner products, or

$$\boxed{g(k)\tilde{u}_n = \tilde{f}_n, \quad n = -N/2, ..., N/2}$$

if the DFT is used to determine the (aliased) coefficients of $f(x)$.

The nodal coefficients of $u_N(x)$ can be computed efficiently via the FFT method.

Fourier Collocation Method

Seek an approximation to $u(x) \in C_p^\infty([0, 2\pi])$ in the modal form

$$u_N(x) = \sum_{k=-N/2}^{N/2} \hat{u}_k e^{ikx}$$

The residual of the equation is then formed

$$\mathcal{R}_N u(x) = \mathcal{L} u_N(x) - f(x) = \sum_{k=-N/2}^{N/2} \hat{u}_k \mathcal{L} e^{ikx} - f(x)$$

Require the residual to vanish point-wise with test functions $\psi_n(x) = \delta(x - x_n)$, we find for $n = -N/2, ..., N/2$

$$\mathcal{R}_N u(x) = \sum_{k=-N/2}^{N/2} (g(k)\hat{u}_k - \tilde{f}) e^{ikx_j} = 0, \quad j = 0, 1, ..., N$$

Fourier Collocation Method

The coefficients of $f(x)$ is determined via interpolation, which implies

$$\tilde{f}_k = \hat{f}_k + \sum_{p>N}^{\infty} \hat{f}_{k+pN}, \quad k = -N/2, ..., N/2$$

As a result, these coefficients are aliased in cases where $\hat{f}_k \neq 0$ for $k > N$.

The Fourier Collocation Method is expressed in terms of algebraic equations for the $(N+1)$ modal coefficients as

$$\boxed{g(k)\tilde{u}_k = \tilde{f}_k, \quad k = -N/2, ..., N/2}$$

Remark: We see that formally, the Fourier Galerkin and Fourier Collocation Methods are distinct, however, the resulting numerical schemes are equivalent in this case.

Fourier Collocation Method

What did we achieve?

- No need for a grid when modal coefficients are unknowns in approximate solution and exact integration can be employed.
- Grids are introduced for discrete evaluation of trigonometric coefficients of $f(x)$ and for IDFT of $\hat{\mathbf{u}}$.
- For linear problems, the modes of the solution is decoupled in spectral space.
- Choice of basis functions meets periodicity assumption/requirement.
- Numerical solution in $\mathcal{O}(N)$ work and efficient $\mathcal{O}(N \log N)$ FFT can be exploited.

Caveat's

- Aliasing errors arises when we need to introduce a grid.
- Choice of basis functions periodic and satisfy BCs.
- We derived numerical schemes for a (simple) linear BVP.

Chebyshev Tau Method

To solve the model problem we seek to represent the discrete solution in terms of Chebyshev polynomials in modal form

$$u_N(x) = \sum_{n=0}^{N} \hat{u}_n T_n(x)$$

where unknown coefficients can be determined via weighted inner products

$$\hat{u}_n = \frac{(u_N, T_n)_w}{(T_n, T_n)_w}$$

by exploiting the orthogonality properties of the basis functions $\{T_n(x)\}_{n=0}^{\infty}$ with respect to the weight $w(x) = (1-x^2)^{-1/2}$.

Assuming homogeneous boundary conditions $u(\pm 1) = 0$ imposes two conditions that must be met by the expansion coefficients

$$\boxed{\sum_{n=0}^{N} \hat{u}_n T_n(\pm 1) = 0}$$

Chebyshev Tau Method

We seek to satisfy the differential equation of the model problem using the weak form

$$\int_{-1}^{1} \mathcal{L}u(x)T_n(x)w(x)dx = \int_{-1}^{1} f(x)T_n(x)w(x)dx, \quad n = 0, 1, ..., N-2$$

or expressed in compact notation

$$(\mathcal{L}u_N, T_n)_w = (f, T_n)_w, \quad n = 0, 1, ..., N-2$$

Recall, arbitrary-order derivatives can be approximated as

$$\frac{d^m u_N}{dx^m} = \sum_{n=0}^{N} \hat{u}_n \frac{d^m T_n(x)}{dx^m} = \sum_{n=0}^{N} \hat{u}_n^{(m)} T_n(x)$$

For the constant coefficient model problem, this translates into

$$\boxed{a\hat{u}_n^{(2)} + b\hat{u}_n^{(1)} + c\hat{u}_n = \hat{f}_n, \quad n = 0, 1, ..., N-2}$$

where $\hat{f}_n$ is the modal coefficients of the projection based on same Chebyshev polynomial basis.

Chebyshev Tau Method

To be able to solve this problem we want to reformulate the problem such that we can determine the unknown coefficients

$$\hat{\mathbf{u}} = (\hat{u}_0, \hat{u}_1, ..., \hat{u}_N)^T$$

from the solution of a linear system of coupled equations

$$\mathcal{A}\hat{\mathbf{u}} = \hat{\mathbf{f}}$$

with the right hand side vector

$$\hat{\mathbf{f}} = (\hat{f}_0, \hat{f}_1, ..., \hat{f}_N)^T$$

consisting of contributions that are known (non-zero right hand side function $f(x)$ and inhomogenous boundary conditions).

QUESTION: How to determine the coefficient matrix $\mathcal{A}$?

Chebyshev Tau Method

The recurrence satisfied by Chebyshev polynomials imposes the following relationships between expansion coefficients and high-order coefficients

$$c_n \hat{u}_n^{(m)} = \hat{u}_{n+2}^{(m)} + 2(n+1)\hat{u}_{n+1}^{(m-1)}, \quad n \geq 0$$

where $c_0 = 2$ and $c_n = 1$ for $n \geq 1$.

$$c_n = \begin{cases} 2 & , n = 0 \\ 1 & , n \geq 1 \end{cases}$$

We can utilize this to rewrite the scheme for the coefficients of the solution such that

$$\hat{u}_n^{(2)} = -\frac{b}{a}\hat{u}_n^{(1)} - \frac{c}{a}\hat{u}_n + \frac{1}{a}\hat{f}_n, \quad n = 0, 1, ..., N-2$$

Then, insert into recurrence (with $m = 2$) and we find

$$c_n \left(-\frac{b}{a}\hat{u}_n^{(1)} - \frac{c}{a}\hat{u}_n + \frac{1}{a}\hat{f}_n \right) = \left(-\frac{b}{a}\hat{u}_{n+2}^{(1)} - \frac{c}{a}\hat{u}_{n+2} + \frac{1}{a}\hat{f}_{n+2} \right) + 2(n+1)\hat{u}_{n+1}^{(1)}, \quad n \geq 0$$

Chebyshev Tau Method

By simplifying last expression we find

$$-\frac{b}{a}\left(c_n\hat{u}_n^{(1)} - \hat{u}_{n+2}^{(1)}\right) - \frac{c}{a}\left(c_n\hat{u}_n - \hat{u}_{n+2}\right) + \frac{1}{a}\left(c_n\hat{f}_n - \hat{f}_{n+2}\right) = 2(n+1)\hat{u}_{n+1}^{(1)}, \quad n \geq 0$$

Again, use recurrence relation for $m = 1$ to rewrite second term

$$-\frac{b}{a}\left((2(n+1)\hat{u}_{n+1}\right) - \frac{c}{a}\left(c_n\hat{u}_n - \hat{u}_{n+2}\right) + \frac{1}{a}\left(c_n\hat{f}_n - \hat{f}_{n+2}\right) = 2(n+1)\hat{u}_{n+1}^{(1)}, \quad n \geq 0$$

Rewriting this expression we find

$$\hat{u}_{n+1}^{(1)} = \frac{1}{2(n+1)}\left(-\frac{b}{a}\left(2(n+1)\hat{u}_{n+1}\right) - \frac{c}{a}\left(c_n\hat{u}_n - \hat{u}_{n+2}\right) + \frac{1}{a}\left(c_n\hat{f}_n - \hat{f}_{n+2}\right)\right), \quad n \geq 0$$

ChebyshevTau Method

Insert last expression into the following recurrence for $m = 1$

$$c_n \hat{u}_n^{(1)} = \hat{u}_{n+2}^{(1)} + 2(n+1)\hat{u}_{n+1}, \quad n \geq 0$$

We setup for $n \geq 3$ the equations for $\hat{u}_n$ such that

$$a_{n,n-1}\hat{u}_{n-1} + a_{n,n}\hat{u}_n + a_{n,n+1}\hat{u}_{n+1} + a_{n,n+2}\hat{u}_{n+2} + a_{n,n+3}\hat{u}_{n+3} = \sum_{m=n-1}^{n+3} \hat{g}_{n,m}\hat{f}_m,$$

The few missing equations for $0 \leq n \leq 2$ can be generated from

$$\hat{u}_n^{(1)} = \frac{2}{\bar{c}_n} \sum_{\substack{p=n+1 \\ p+n \text{ odd}}}^{\infty} p\hat{u}_p, \quad \hat{u}_n^{(2)} = \frac{1}{\bar{c}_n} \sum_{\substack{p=n+2 \\ p+n \text{ even}}}^{\infty} p(p^2 - n^2)\hat{u}_p, \quad n \geq 0$$

where $\bar{c}_n = 1 + \delta_{n0}$ with δ_{ij} Kronecker's delta function.

Remark: By instead using these two latter expressions for rewriting the scheme for all $0 \leq n \leq N-2$ the coefficient matrix of the system is dense above the diagonal.

Chebyshev Tau Method

Therefore, by using the recurrence relationships we can form a sparse linear system of algebraic equations of the form

$$A\hat{\mathbf{u}} = \hat{\mathbf{g}}, \quad A \in \mathbb{R}^{(N+1)\times(N+1)}$$

where discrete representation of two boundary conditions are included and $\hat{\mathbf{g}}$ contains elements which are linear combinations of elements in $\hat{\mathbf{f}}$.

The discrete solution $\mathbf{u}$ can then be obtained from $\tilde{\mathbf{u}}$ efficiently with $\mathcal{O}(N \log N)$ effort using the inverse FFT method for Chebyshev polynomials if the Gauss-Lobatto nodes are chosen for the evaluation of $\mathbf{u}$.

Fast Chebyshev Transform (FCT)

The Fast Chebyshev Transform (FCT) is the mapping between the discrete set of real value $f(x_j) = f_j$, x_j the Gauss-Lobatto nodes and the real coefficients $\tilde{f}_k$ of the Chebyshev polynomial expansion.

In summary, the $(N+1)$-point FCT is defined as

$$\tilde{f}_k = \frac{2}{N\bar{c}_k} \sum_{j=0}^{N} \frac{1}{\bar{c}_j} f_j \cos\left(\frac{\pi j k}{N}\right), \quad k = 0, 1, ..., N$$

where $\bar{c}_k = 1 + \delta_{k0} + \delta_{kN}$ and δ_{ij} is Kronecker Delta.. The Inverse Fast Chebyshev Transform (IFCT) is

$$f_j = \sum_{k=0}^{N} \tilde{f}_k \cos\left(\frac{\pi j k}{N}\right), \quad j = 0, 1, ..., N$$

The FCT and IFCT operations can be computed efficiently using the Fast Fourier Transform (FFT) method with $\mathcal{O}(N \log N)$ operations.

Fast Chebyshev Transform via FFT in Matlab

From the expression for the IFCT it is clear that the discrete Chebyshev polynomial expansion can be considered a Discrete Cosine Transform (DCT). The FFT can be employed in three steps

- Introduce a continuous even and periodic extension which turns an N-point into a $2N$-point function representation.

$$\begin{bmatrix} f_N & f_{N-1} & \cdots & f_1 & f_2 & \cdots & f_{N-1} \end{bmatrix}$$

  Remark: the Matlab `fft` command assumes reverse ordering.

- Employ the FFT to determine the $2N$ complex coefficients of the complex Fourier expansion with ordering (in Matlab)

$$\begin{bmatrix} \tilde{c}_0 & | & \tilde{c}_1 & \tilde{c}_2 & \cdots & \tilde{c}_{N-1} & | & \tilde{c}_{-N} & \tilde{c}_{N-1} & \cdots & \tilde{c}_{-1} \end{bmatrix}$$

- Compute real trigonometric coefficients for cosine terms from

$$\begin{aligned} a_0 &= \tilde{c}_0 \\ a_n &= \tilde{c}_n + \tilde{c}_{-n} = 2\tilde{c}_n, \quad n = 1, 2, ..., N-1 \\ a_N &= 2\tilde{c}_{-N} \end{aligned}$$

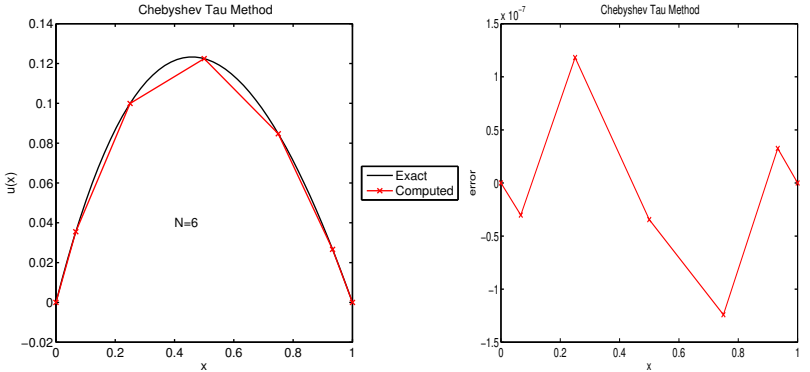
Fast Chebyshev Transform Routines via FFT in Matlab

```
{\blue \bf function uh = fct(u)}
% Fast Chebyshev Transform
%
%    uh : Discrete Chebyshev Transform Coefficients.
%    u  : Function values evaluated at Chebyshev Gauss Lobatto nodes
%         with nodes ordered increasingly x_i=[-1,...,1] for i=1,2...,N
%
N  = length(u);
u  = ifft([u([N:-1:1 2:N-1])]); % reverse ordering due to Matlab's fft
uh = ([u(1); 2*u(2:(N-1)); u(N)]);
return

{\blue \bf function u = ifct(uh)}
% Inverse Fast Chebyshev Transform
%
%    uh : Discrete Chebyshev Transform Coefficients.
%    u  : Function values evaluated at Chebyshev Gauss Lobatto nodes
%         with nodes ordered increasingly x_i=[-1,...,1] for i=1,2...,N
%
N = length(uh);
u=fft([uh(1); [uh(2:N-1); uh(N)*2; uh(N-1:-1:2)]*0.5]);
u=(u(N:-1:1)); % reverse ordering due to Matlab's fft.
u = real(u);
return
```

Chebyshev Tau Method

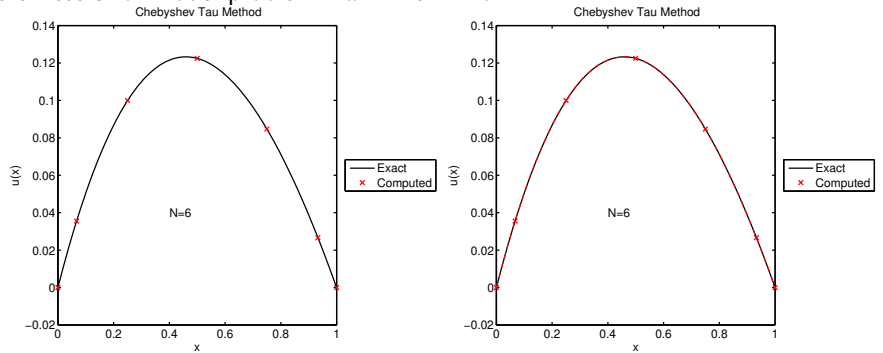
Parameters for model problem : $a = -c = -b = 1$



BAD VISUALIZATION! - WHY?

Tau Method

Parameters for model problem : $a = -c = -b = 1$



Left: Nodal values at collocation points.
Right: Interpolated values for representing entire global solution.

GOOD VISUALIZATIONS! - WHY?

ChebyshevTau Method

What did we achieve?

- No constraints on choice of basis functions except orthogonality. (Convenience and generality)
- Sparse system for unknown coefficients generated through tedious manipulations. (Direct sparse solvers)
- Relied on continuous transformations and constant-coefficient differential equation. (Simple problem)
- By choice of Chebyshev basis efficient $\mathcal{O}(N \log N)$ transform to physical function values via FTT possible.
- Easy to incorporate boundary conditions.

Caveat's

- Coefficients of differential equations not constant in general (destroys sparsity).
- In general, coefficients of $f(x)$ cannot be determined exactly.
- For general choices of basis functions, we need $\mathcal{O}(N^2)$ transform methods.

Spectral Polynomial Collocation Method

To solve the model problem we seek to represent the discrete solution in terms of polynomials

$$u_N(x) = \sum_{n=0}^{N} \hat{u}_n P_n^{(\alpha)}(x) = \sum_{n=0}^{N} u_n h_n(x)$$

To enforce the residual at interior collocation nodes

$$\mathcal{R}u_N(x_i) = 0, \quad i = 1, 2..., N-1$$

together with imposing homogenous Dirichlet boundary conditions

$$u_N(x_i) = 0, \quad i = 0, N$$

we need to introduce a grid.

It is convenient to introduce a set of Jacobi Gauss-Lobatto grid points for ultraspherical polynomials

$$x_i = \{x : -1 \leq x \leq 1 \mid (1-x^2)\tfrac{d}{dx} P_N^{(\alpha)} = 0 \ \vee \ i = 0, 1, ..., N\}$$

which includes nodes at the boundaries.

Spectral Polynomial Collocation Method

We seek to satisfy the differential equation at interior points. To do this, we introduce an approximation of the operator, e.g.

$$\mathcal{L}_N u_N = -\mathcal{D}_N \left(a \frac{du_N}{dx} \right) + b \frac{du_N}{dx} + c$$

Recall, arbitrary-order derivatives can be approximated as

$$\frac{d^m}{dx^m} u_N(x) = \sum_{n=0}^{N} u_i \frac{d^m}{dx^m} h_n(x)$$

The first term on the right assumes that $a = a(x) \neq$constant in which case

$$\mathcal{D}v = \frac{d}{dx} \mathcal{I}_N v$$

denotes the interpolation derivative of the function v.

Spectral Polynomial Collocation Method

Thus, the collocation equations on the set of nodes are

$$\mathcal{L}_N u_N(x_i) = f(x_i), \quad 1 \leq i \leq N-1$$
$$u_N(x_i) = 0, \quad i = 0, N$$

In matrix terms it can be expressed in terms of the discrete operator (BCs not enforced yet)

$$\mathcal{L}_N = -\mathcal{D}\mathcal{A}\mathcal{D} + \mathcal{B}\mathcal{D} + \mathcal{C}$$

with $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ diagonal matrices which holds the possible variable values of a, b and c. For example

$$\mathcal{A} = \text{diag } \mathbf{a}, \quad \mathbf{a} = (a(x_i))_{i=0,1,\ldots,N}$$

$\mathcal{D}$ is the standard nodal interpolation derivative matrix based on Jacobi Gauss-Lobatto nodes.

Remark: If $a(x) =$ contant the operator can be computed from

$$\mathcal{L}_N = -a\mathcal{D}^2 + \mathcal{B}\mathcal{D} + \mathcal{C}$$

Spectral Polynomial Collocation Method

As a result we can form the linear system

$$\mathcal{L}_N \mathbf{u} = \mathbf{f}$$

which do not yet take into account boundary conditions.

To enforce boundary conditions, we need to modify the discrete matrix operator $\mathcal{L}_N$. Two options

1. Eliminate equations for known coefficients
   - Reduce size of linear system by the two boundary equations and corresponding two unknown grid values u_0 and u_N by discarding equations for the Dirichlet conditions. Move known terms for interior equations to right hand side.
2. Include equations for known coefficients
   - Modify first and last row of linear system and right hand side vector to include boundary conditions.

After which we can solve the modified system

$$\tilde{\mathcal{L}}_N \mathbf{u} = \tilde{\mathbf{f}}$$

Spectral Polynomial Collocation Method

What did we achieve?

- No constraints on choice of basis functions.
  (Convenience and generality)
- Choice of grid points (almost) free. (Generality)
- Non-constant coefficients can be handled in a straightforward way. (Generality)
- Relied on no transformations. (Generality)
- Easy to incorporate boundary conditions.

Caveat's

- Dense system for unknown coefficients generated in a straightforward way but effort is not immediately scalable.
  (Expensive direct solvers, some tricks with iterative solvers)

Boundary Value Problems

To summarize, for the solution of (time-independent) Boundary Value Problems (BVPs) we need to choose

- Basis set
  - Periodic problems $\rightarrow$ Trigonometric polynomials (Fourier)
  - Non-periodic problems $\rightarrow$ Legendre/Chebyshev polynomials
- Number of modes or grid nodes
- MWR/Numerical method
  - Galerkin/Tau
  - Collocation
- How to impose boundary conditions
  - Explicitly via incorporating conditions through algebraic constraints
  - Implicitly via choice of basis set

Remarks on solving BVPs

Words of caution

- For $N \to \infty$ the discrete nodal spectral operators will be increasingly ill-conditioned (impact accuracy) for N large.
- Direct Gaussian Elimination is $\mathcal{O}(N^3)$ for dense operators and may become inefficient if N is large.

Conclusion: Work effort for the solution of general BVPs with Spectral Methods not immediately $\mathcal{O}(N)$ or $\mathcal{O}(N \log N)$ and therefore not scalable.

Thus, for these reasons single-domain spectral methods are not widely used for very large problems, but rather as components in advanced solvers (fx. DG-FEM).

However, we have seen

- Easy proto-typing of single domain spectral solvers.
- High accuracy with few unknown possible for problems with sufficient smoothness.

Verification

Verification is the process of testing for correctness of an implementation by finding and removing mistakes in source code. There are two essential parts of the verification process

- Verification of the method (solution verification)
- Verification of the implemented method (code verification)

Two widely used approaches to verify correctness is to use

- Method of Manufactured Solutions (choose solution), or
- Use exact solution to model equations.

as a basis for convergence tests towards grid-independent solution. It is of interest to assess the

- accuracy of input data
- numerical solution error
- accuracy of output data

It is recommended to test parts of code separately in small steps.

Ideas for verification

Due to global nature of Spectral Methods small bugs can influence all of the solution.

Examples of useful verification tests

- Check that high-mode coefficients in modal expansions decay (exponentially fast) when $N \to \infty$.
- Check that coefficients in modal expansion are not all of same magnitude for any N (under resolved, aliased, polluted).
- Do a convergence tests to verify asymptotic order of accuracy.
- Construct solution to problems via Method of Manufactured Solutions and verify implemented routines.
- Always visualize solutions, magnitude of coefficients, etc. to try and spot where things goes wrong or if things looks ok.

Multiple dimensions

For the solution of boundary value problem in multiple domains we follow the same conceptual steps as we have done in 1D.

For example, to represent the (square-integrable) solution $u(x, y)$ in two dimensions $(x, y) \in [-1, 1]^2$, we can use a finite spectral series of the form (blend of ultraspherical polynomials)

$$u_N(x, y) = \sum_{i=0}^{N} \sum_{j=0}^{N} \hat{u}_{ij} P_i^{(\alpha)}(x) P_j^{(\beta)}(y) = \sum_{i=0}^{N} u_{ij} h_i(x) h_j(y)$$

Restrict attention to the construction of collocation methods and require the residual to be enforced at interior collocation nodes

$$\mathcal{R}u_N(x_i) = 0, \quad (i, j) \in \{(x_i, y_i) : (x_i, y_i) \in \Omega([-1, 1])\}$$

together with imposing boundary conditions

$$\mathcal{B}u_N(x_i, y_i) = 0, \quad (i, j) \in \{(x_i, y_i) : (x_i, y_i) \in \partial\Omega([-1, 1])\}$$

As usual, we need to introduce a grid.

Multiple dimensions

It is convenient to introduce a set of Jacobi Gauss-Lobatto grid points for ultraspherical polynomials

$$x_i = \{x : -1 \leq x \leq 1 \mid (1-x^2)\tfrac{d}{dx}P_N^{(\alpha)} = 0 \ \vee \ i = 0, 1, ..., N\}$$

which includes nodes at the boundaries.

This provide a basis for generating basis functions via tensor products, e.g. we can introduce the set of nodal basis functions

$$h_{i+(N+1)j}(x, y) = h_i(x)h_j(y), \quad i, j = 0, 1, ..., N$$

which fulfills the interpolating properties

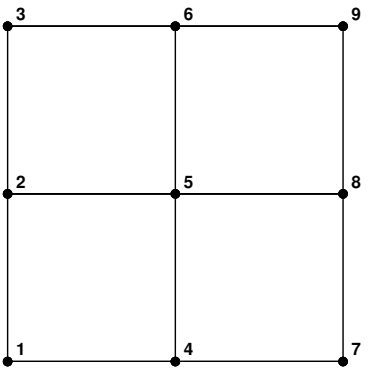
$$h_n(x_m, y_m) = \delta_{mn}, \quad n = i + (N+1)j$$

where a single index n has been introduced to replace the double index (i, j) for the basis functions.

Multiple dimensions

```
>> x = linspace(-1,1,3);
>> y = linspace(-1,1,3);
>> [X,Y] = meshgrid(x,y);
X =
  -1     0     1
  -1     0     1
  -1     0     1
Y =
  -1    -1    -1
   0     0     0
   1     1     1
 >> u = sin(pi*X).*cos(pi*Y);

 %% Useful Matlab tricks
 >> u2 = u(:); % easy conversion to a vector
 >> u(:) = u2(:); % easy conversion back to array
```



Multiple dimensions

To develop global spectral differentiation matrices for approximation of differential operations such as

$$\frac{du_N(x,y)}{dx} = \mathcal{D}_x \mathbf{u}, \quad \frac{du_N(x,y)}{dy} = \mathcal{D}_y \mathbf{u}$$

for use with a solution vector $\mathbf{u}$ that are represented on tensor product grids. It is noted, that $\mathbf{u}$ contains nodal values of the solution ordered in column major order in Matlab.

To construct global operators in multiple dimensions, we can use *Kronecker products* ($\otimes$). Conveniently, these can be computed in Matlab using `kron`.

The Kronecker product C of two matrices A and B is denoted by $C = A \otimes B$ and is computed by the command

```
>> C = kron(A,B);
```

If $A \in \mathbb{R}^{p \times q}$ and $B \in \mathbb{R}^{r \times s}$ then $C \in \mathbb{R}^{pr \times qs}$.

Multiple dimensions

$$I = \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}, \quad D = \begin{bmatrix} -\frac{3}{2} & 2 & -\frac{1}{2} \\ -\frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & -2 & \frac{3}{2} \end{bmatrix},$$

$$D_y = I \otimes D = \left[\begin{array}{ccc|ccc|ccc} -\frac{3}{2} & 2 & -\frac{1}{2} & & & & & & \\ -\frac{1}{2} & 0 & \frac{1}{2} & & & & & & \\ \frac{1}{2} & -2 & \frac{3}{2} & & & & & & \\ \hline & & & -\frac{3}{2} & 2 & -\frac{1}{2} & & & \\ & & & -\frac{1}{2} & 0 & \frac{1}{2} & & & \\ & & & \frac{1}{2} & -2 & \frac{3}{2} & & & \\ \hline & & & & & & -\frac{3}{2} & 2 & -\frac{1}{2} \\ & & & & & & -\frac{1}{2} & 0 & \frac{1}{2} \\ & & & & & & \frac{1}{2} & -2 & \frac{3}{2} \end{array}\right]$$

Multiple dimensions

$$D_x = D \otimes I = \left[\begin{array}{ccc|ccc|ccc} -\frac{3}{2} & & & 2 & & & -\frac{1}{2} & & \\ & -\frac{3}{2} & & & 2 & & & -\frac{1}{2} & \\ & & -\frac{3}{2} & & & 2 & & & -\frac{1}{2} \\ \hline -\frac{1}{2} & & & & & & \frac{1}{2} & & \\ & -\frac{1}{2} & & & & & & \frac{1}{2} & \\ & & -\frac{1}{2} & & & & & & \frac{1}{2} \\ \hline \frac{1}{2} & & & -2 & & & \frac{3}{2} & & \\ & \frac{1}{2} & & & -2 & & & \frac{3}{2} & \\ & & \frac{1}{2} & & & -2 & & & \frac{3}{2} \end{array}\right]$$

Multiple dimensions

Consider the BVP

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y), \quad (x,y) \in [-1,1]^2$$
$$u = 0$$

Using a collocation method we can construct the operator for the left hand side using global operators as

$$\mathcal{L} = \mathcal{D}_x^2 + \mathcal{D}_y^2$$

The right hand side vector is constructed in the straightforward way as

$$\mathbf{f} = (f(x_1,y_1), f(x_2,y_2), ..., f(x_M,y_M))^T, \quad M = N_x N_y$$

where N_x and N_y are the nodes in each Cartesian direction.

By modification of $\mathcal{L}$ and $\tilde{\mathbf{f}}$ to impose boundary conditions

$$\tilde{\mathcal{L}}\mathbf{u} = \tilde{\mathbf{f}}$$

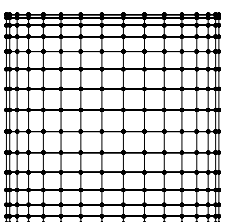
which we can solve for $\mathbf{u} = \mathcal{L}^{-1}\tilde{\mathbf{f}}$.

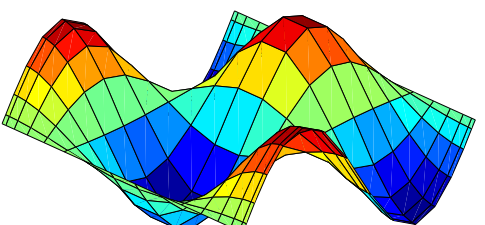
Multiple dimensions

Conceptual solution of Spectral 2D BVP

```
% allocate storage for solution and rhs vector
>> u = X*0;
% solve BVP via direct method
>> u(:) = Lt\ft(:);
% Visualize solution (surface of right most plot)
>> surf(X,Y,u)
```

Assumed Lt and $\mathbf{ft}$ defined in Matlab workspace.





Summary

With the fundamentals in place, we have building stones that are useful for

- For generic construction of high-order Spectral Methods.
- That can be used in extensions in higher dimensional approximations.

Some reasons for interests into Spectral Methods for BVPs are

- Accuracy (for smooth problems) with few unknowns and easy estimation.
- Efficiency if
  - N is large and FFT can be exploited (Fourier).
  - N is small and dense well-conditioned operations can be exploited to deliver high accuracy with few degrees of freedom (Chebyshev/Legendre), although not scalable for $N \to \infty$.
- Global nature allow for accurate representation of solutions.
- Proto-typing of problems can be straightforward and useful as a part of model verification purposes.