1. Multigrid Idea No. 1: Use Coarse Grids

starting point: Fourier mode analysis of the errors

- decompose the error $e^{(i)}$ into its Fourier components $\sin(k\pi x_j)$ (Fourier transform)
- observe how they change/decrease under a standard relaxation like Jacobi or Gauss-Seidel:
  - The high frequency part (with respect to the underlying grid) is reduced quite quickly.
  - The low frequency part (w.r.t. the grid) decreases only very slowly; actually the slower, the finer the grid is.

Multigrid Idea No. 1:

- on a sufficiently coarse grid, even very low frequencies are oscillatory with respect to the mesh size

⇒ use multiple grids to solve the system of equations
2. Multigrid Idea No. 2: Coarse Grid Correction

Solving the problem on a coarser grid

- will be comparably (very) fast
- will give us a good initial guess (nested iteration, “poor man’s multigrid”)
- unfortunately, will not improve a fine grid solution any further

⇒ Multigrid Idea No. 2: use the residual equation:

- solve the residual equation $Ae = r$ on a coarser grid
- will lead to an approximation of the error $e$
- add this approximation to the current fine-grid solution

Required: transfer between coarse and fine grid

- restriction of the residual to the coarse grid
- interpolation of the correction to the fine grid
3. Towards Multigrid

• to solve on the coarse grid, we can again use the idea of coarse grid correction

⇒ sequence of equidistant grids on our domain:

\[ \Omega_l, \quad l = 1, 2, \ldots, L, \]

with mesh width \( h_l = 2^{-l} \)

• linear system of equations \( A_l x_l = b_l \) on each level

• multigrid algorithm will result from recursive application of a two-grid correction scheme
4. A Two-Grid Method

4.1. Algorithm

1. **relaxation/smoothing** on the fine level system $\Rightarrow$ solution $x_l$
2. compute the **residual** $r_l = b_l - A_l x_l$
3. **restriction** of $r_l$ to the coarse grid $\Omega_{l-1}$
4. compute a **solution** to $A_{l-1} e_{l-1} = r_{l-1}$
5. **interpolate** the coarse grid solution $e_{l-1}$ to the fine grid $\Omega_l$
6. add the resulting **correction** to $x_l$
7. again, **relaxation/smoothing** on the fine grid (if necessary)
4.2. Components of a Correction Scheme

**pre-smoothing:** reduce the high-frequency error components, and get a smooth error

**restriction:** transfer residual from fine grid to coarse grid, for example by

- **injection:** inherit the coarse grid values and forget the others
- **(full) weighting:** apply some averaging process

**coarse grid correction:** provide an (approximate) solution on the coarse grid (direct, if coarse enough; some iterative scheme otherwise)

**interpolation:** transfer coarse grid solution/correction from coarse grid to fine grid, for example by linear interpolation

**post-smoothing:** to remove new high-frequency error components that might be produced by the coarse grid correction and interpolation
5. **The Multigrid V-Cycle**

If we coarse grid correction to solve the coarse grid system, the resulting recursive algorithmic scheme is called a *multigrid V-cycle*:

1. smoothing on the fine level system $\Rightarrow$ solution $x_l$
2. compute the residual $r_l = b_l - A_l x_l$
3. restriction of $r_l$ to the coarse grid $\Omega_{l-1}$
4. **solve coarse grid system** $A_{l-1} e_{l-1} = r_{l-1}$ **by a recursive call to the V-cycle algorithm**
5. interpolate the coarse grid solution $e_{l-1}$ to the fine grid $\Omega_l$
6. add the resulting correction to $x_l$
7. **post-smoothing** on the fine grid (if necessary)

**Implementation:**

- on the coarsest grid: direct solution
- number of smoothing steps is typically very small (1 or 2)
6. More multigrid schemes

6.1. The W-cycle

perform **two** coarse grid correction steps instead of one

(V-cycle and W-cycle)

- more expensive
- useful in situations where the coarse grid correction is not very accurate
6.2. FMV-cycle (Full Multigrid V-cycle)

Recursive algorithm:

- perform an **FMV-cycle** on the next coarser grid to get a good initial solution
- interpolate this initial guess to the current grid
- perform a **V-cycle** to improve the solution

For the Poisson equation, a single FMV-cycle can be sufficient to solve the equation (to the level of discretization).
7. **Basic Convergence Results**

**Cost (storage and computing time):**

- **1D:** \(c \cdot n + c \cdot n/2 + c \cdot n/4 + c \cdot n/8 + \ldots \leq 2c \cdot n = O(n)\)
- **2D:** \(c \cdot n + c \cdot n/4 + c \cdot n/16 + c \cdot n/64 + \ldots \leq 4/3c \cdot n = O(n)\)
- **3D:** \(c \cdot n + c \cdot n/8 + c \cdot n/64 + c \cdot n/512 + \ldots \leq 8/7c \cdot n = O(n)\)
- overall costs are dominated by the costs of the finest grid

**Speed of convergence:**

- significant acceleration compared with relaxation methods (if all components are chosen carefully)
- \textit{“textbook multigrid efficiency”}: \(\|e^{(m+1)}\| \leq \gamma \|e^{(m)}\|\), where convergence rate \(\gamma < 1\) is independent of the number of unknowns
  \[\Rightarrow\] constant number of multigrid steps to obtain a given number of digits
  \[\Rightarrow\] overall computational work increases only linearly with the number of unknowns

observed for Poisson type problems (and several others)