High-order multilevel optimization strategies and their application to the training of artificial neural networks

E. Riccietti (IRIT-INP, Toulouse)

Joint work with: H. Calandra (Total)
S. Gratton (IRIT-INP, Toulouse)
X. Vasseur (ISAE-SUPAERO, Toulouse)

Manchester- 6 November 2018

Context

We consider large-scale nonlinear unconstrained optimization problems:

$$\min_x f(x)$$

Context

We consider large-scale nonlinear unconstrained optimization problems:

$$\min_x f(x)$$

Classical iterative optimization methods:

$$f(x_k + s) \simeq T_q(x_k, s)$$

with $T_q(x_k, s)$ Taylor model of order q. At each iteration we compute a step s_k to update the iterate:

$$\min_s m_k(x_k, s) = T_q(x_k, s) + \frac{\lambda_k}{q+1}\|s\|^{q+1}, \qquad \lambda_k > 0$$

Classical choices

- Least-squares, $f = \|F\|^2$: Levenberg-Marquardt (LM), $q = 1$,
  $T_1(x_k, s) = f(x_k) + s^T \nabla f(x_k) + \frac{1}{2} s^T B_k s$,
  $\nabla^2 f(x_k) \sim B_k = J(x_k)^T J(x_k)$, J Jacobian matrix of F

- Adaptive Cubic Regularization method (ARC), $q = 2$.
  $T_2(x_k, s) = f(x_k) + s^T \nabla f(x_k) + \frac{1}{2} s^T \nabla^2 f(x_k) s$

  C.Cartis, N. Gould, Ph. Toint, Adaptive cubic regularisation methods for unconstrained optimization, 2009

Extension to higher-order methods ($q > 2$)

Worst-case evaluation complexity for unconstrained nonlinear optimization using high-order regularized models, E. G. Birgin, J. L. Gardenghi, J. M. Martnez, S. A. Santos and Ph. L. Toint, 2017

Unifying framework for global convergence and worst-case complexity is presented.

☺ better complexity
☹ needs higher-order derivatives, model is expensive to minimize

Subproblem solution

Solving

$$\min_s T_q(x_k, s) + \frac{\lambda_k}{q+1}\|s\|^{q+1}$$

represents greatest cost per iteration, which depends on the size of the problem.

$$\Downarrow$$

 Multilevel trust region method, Gratton, Sartenaer, Toint, 2008

Hierarchy of problems

- $\{f_l(x_l)\}$, $x_l \in \mathcal{D}_l$
- $|\mathcal{D}_l| < |\mathcal{D}_{l+1}|$
- f_l is cheaper to optimize compared to f_{l+1}

Our contributions

E. G. Birgin, J. L. Gardenghi, J. M. Martinez, S. A. Santos and Ph. L. Toint, 2017

- one level methods: non-scalable

Gratton, Sartenaer, Toint, 2008

- method for second order models
- used just for problems with a geometrical structure

$$\Downarrow$$

We propose a family of scalable multilevel methods using high-order models.

Our contributions

E. G. Birgin, J. L. Gardenghi, J. M. Martinez, S. A. Santos and Ph. L. Toint, 2017

- one level methods: non-scalable

Gratton, Sartenaer, Toint, 2008

- method for second order models
- used just for problems with a geometrical structure

$$\Downarrow$$

We propose a family of scalable multilevel methods using high-order models.

- Part I: multilevel extension of iterative high-order optimization methods

Our contributions

E. G. Birgin, J. L. Gardenghi, J. M. Martinez, S. A. Santos and Ph. L. Toint, 2017

- one level methods: non-scalable

Gratton, Sartenaer, Toint, 2008

- method for second order models
- used just for problems with a geometrical structure

$$\Downarrow$$

We propose a family of scalable multilevel methods using high-order models.

- Part I: multilevel extension of iterative high-order optimization methods
- Part II: use of the multilevel methods for the training of artificial neural network (no underlying geometry)

Part I

1. Multilevel extension of iterative high-order optimization methods

Multilevel setting

- At each level l, $x \in \mathbb{R}^{n_l}$. $l_{\max}$ finest level, 0 coarsest level.

level $l_{\max}$	$\mathbb{R}^n$	$x^{l_{\max}}$	$f^{l_{\max}} = f$	$\mu^{l_{\max}} = f$
$\vdots$		$\vdots$	$\vdots$	$\vdots$
level $l+1$	$\mathbb{R}^{n_{l+1}}$	x^{l+1}	f^{l+1}	μ^{l+1}
	$R^{l+1} \Downarrow \quad \Uparrow P^{l+1}$			
level l	$\mathbb{R}^{n_l}$	x^l	f^l	μ^l
$\vdots$		$\vdots$	$\vdots$	$\vdots$
level 0	$\mathbb{R}^{n_0}$	x^0	f^0	μ^0

- f^l represents f on the coarse spaces (it is e.g. the discretization of f on a coarse space)
- The functions μ^l are modifications of the f^l to ensure inter-level coherence.

Coherence between levels, $q = 1$

Lower level model:

- Let $x_0^{l-1} = Rx_k^l$. Model with first order correction:

$$\mu^{l-1} = T_q^{l-1}(x_0^{l-1}, s^{l-1}) + (R^l \nabla f^l(x_k^l) - \nabla f^{l-1}(x_k^{l-1}))^T s^{l-1}$$

This ensures that

$$\nabla \mu^{l-1}(x_0^{l-1}) = R^l \nabla f^l(x_k^l)$$

$\rightarrow$ first-order behaviours of f^l and μ^{l-1} are coherent in a neighbourhood of the current approximation. If $s^l = P^l s^{l-1}$

$$\nabla f^l(x_k^l)^T s^l = \nabla f^l(x_k^l)^T P^l s^{l-1} = \nabla \mu^{l-1}(x_0^{l-1})^T s^{l-1}.$$

Coherence between levels

Lower level model: Let $x_0^{l-1} = Rx_k^l$. We define $\mu_{q,k}^{l-1}$ as

$$\mu_{q,k}^{l-1}(x_{0,k}^{l-1} + s^{l-1}) = T_q^{l-1}(x_{0,k}^{l-1} + s^{l-1}) +$$
$$\sum_{i=1}^{q} \frac{1}{i!}[(\nabla^i f^l \underbrace{\otimes R \otimes \cdots \otimes R}_{i \text{ times}})(x_k^l) - \nabla^i f^{l-1}(x_{0,k}^{l-1})] \underbrace{(s^{l-1}, \ldots, s^{l-1})}_{i \text{ times}}.$$

It holds: $\nabla^i \mu_{q,k}^{l-1}(x_{0,k}^{l-1}) = (\nabla^i f^l \underbrace{\otimes R \otimes \cdots \otimes R}_{i \text{ times}})(x_k^l), \qquad i = 1, \ldots, q.$

$\rightarrow$ behaviours of f^l and μ^{l-1} are coherent up to order q in a neighbourhood of the current approximation.

$$\nabla^i f^l(x_k^l, \overbrace{s^l, \ldots, s^l}^{i \text{ times}}) = \nabla^i f^l(x_k^l, Ps^{l-1}, \ldots, Ps^{l-1}) =$$
$$(\nabla^i f^l \underbrace{\otimes R \otimes \cdots \otimes R}_{i \text{ times}})(x_k^l, s^{l-1}, \ldots, s^{l-1}) = \nabla^i m_{q,k}^{l-1}(x_{0,k}^{l-1}, s^{l-1}, \ldots, s^{l-1}).$$

One level strategy

At level $l = 1$, let x_k^l be the current approximation. We look for a correction s_k^l to define the new approximation $x_{k+1}^l = x_k^l + s_k^l$.

x_k^l

One level strategy

At level $l = 1$, let x_k^l be the current approximation. We look for a correction s_k^l to define the new approximation $x_{k+1}^l = x_k^l + s_k^l$.

$$x_k^l \xrightarrow{T_q^l} x_{k+1}^l = x_k^l + s_k^l$$

Multilevel strategy

Two choices:

1. minimize regularized Taylor model, get s_k^l,
2. choose lower level model μ^{l-1}:

$$x_k^l$$

Multilevel strategy

Two choices:

1. minimize regularized Taylor model, get s_k^l,
2. choose lower level model μ^{l-1}:

$$x_k^l \xrightarrow{T_q^l} x_{k+1}^l = x_k^l + s_k^l$$

Multilevel strategy

Two choices:

1. minimize regularized Taylor model, get s_k^l,
2. choose lower level model μ^{l-1}: if $\|R\nabla f^l(x_k^l)\| \geq \kappa \|\nabla f^l(x_k^l)\|$

x_k^l

Multilevel strategy

Two choices:

1. minimize regularized Taylor model, get s_k^l,
2. choose lower level model μ^{l-1}: if $\|R\nabla f^l(x_k^l)\| \geq \kappa \|\nabla f^l(x_k^l)\|$

$$x_k^l$$
$$\Big\downarrow R^l$$
$$R^l x_k^l := x_0^{l-1}$$

Multilevel strategy

Two choices:

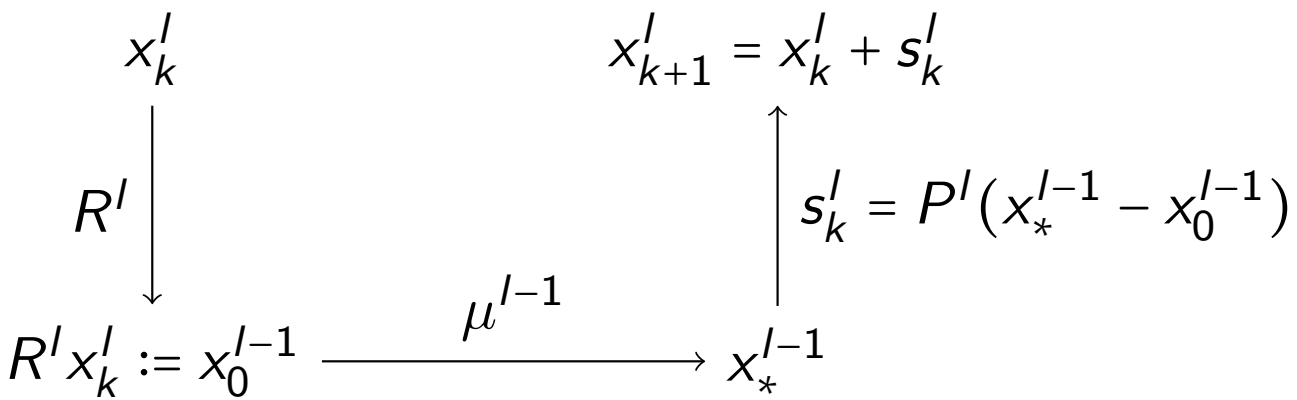
1. minimize regularized Taylor model, get s_k^l,
2. choose lower level model μ^{l-1}: if $\|R\nabla f^l(x_k^l)\| \geq \kappa \|\nabla f^l(x_k^l)\|$

$$
\begin{array}{ccc}
x_k^l & & \\
\Big\downarrow R^l & & \\
R^l x_k^l := x_0^{l-1} & \xrightarrow{\quad\mu^{l-1}\quad} & x_*^{l-1}
\end{array}
$$

Multilevel strategy

Two choices:

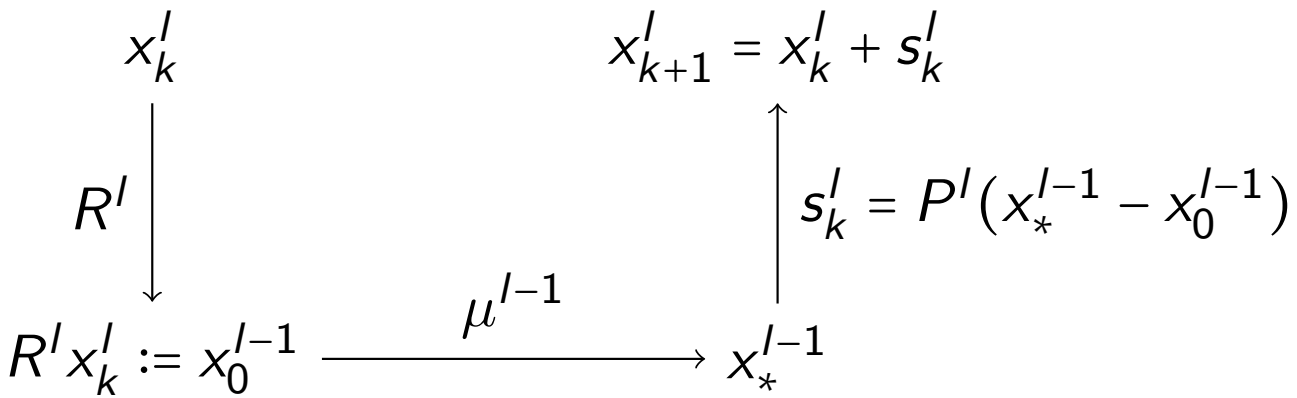
1. minimize regularized Taylor model, get s_k^l,
2. choose lower level model μ^{l-1}: if $\|R\nabla f^l(x_k^l)\| \geq \kappa \|\nabla f^l(x_k^l)\|$



Multilevel strategy

Two choices:

1. minimize regularized Taylor model, get s_k^l,
2. choose lower level model μ^{l-1}: if $\|R\nabla f^l(x_k^l)\| \geq \kappa \|\nabla f^l(x_k^l)\|$



- The lower level model is cheaper to optimize.
- The procedure is recursive: more levels can be used.

Theoretical results: global convergence

Assumption

Let us assume that for all l the q-th derivative tensors of f^l are Lipschitz continuous.

Theorem

Let Assumption hold. Assume that the number of fine steps taken is not smaller than the number of lower level steps taken. Then, the sequence of iterates generated by the algorithm converges globally to a first-order stationary point.

Theoretical results: global convergence

Assumption

Let us assume that for all l the q-th derivative tensors of f^l are Lipschitz continuous.

Theorem

Let Assumption hold. Assume that the number of fine steps taken is not smaller than the number of lower level steps taken. Then, the sequence of iterates generated by the algorithm converges globally to a first-order stationary point.

E. G. Birgin, J. L. Gardenghi, J. M. Martnez, S. A. Santos and Ph. L. Toint, 2017: generalized to multilevel framework

Gratton, Sartenaer, Toint, 2008: extended to higher-order models and simplified

Theoretical results: complexity

Theorem

Let Assumption hold. Let f_{low} be a lower bound on f. Then, the method requires at most

$$K_3 \frac{(f(x_{k_1}) - f_{low})}{\epsilon^{\frac{q+1}{q}}} \left(1 + \frac{|\log \gamma_1|}{\log \gamma_3}\right) + \frac{1}{\log \gamma_3} \log\left(\frac{\lambda_{\max}}{\lambda_0}\right)$$

iterations to achieve an iterate x_k such that $\|\nabla f(x_k)\| \leq \epsilon$, where

$$K_3 := \frac{q+1}{\eta_1 \lambda_{\min}} \max\{K_1^{1/q}, K_2^{1/q}\}.$$

Theoretical results: complexity

Theorem

Let Assumption hold. Let f_{low} be a lower bound on f. Then, the method requires at most

$$K_3 \frac{(f(x_{k_1}) - f_{low})}{\epsilon^{\frac{q+1}{q}}} \left(1 + \frac{|\log \gamma_1|}{\log \gamma_3}\right) + \frac{1}{\log \gamma_3} \log\left(\frac{\lambda_{\max}}{\lambda_0}\right)$$

iterations to achieve an iterate x_k such that $\|\nabla f(x_k)\| \leq \epsilon$, where

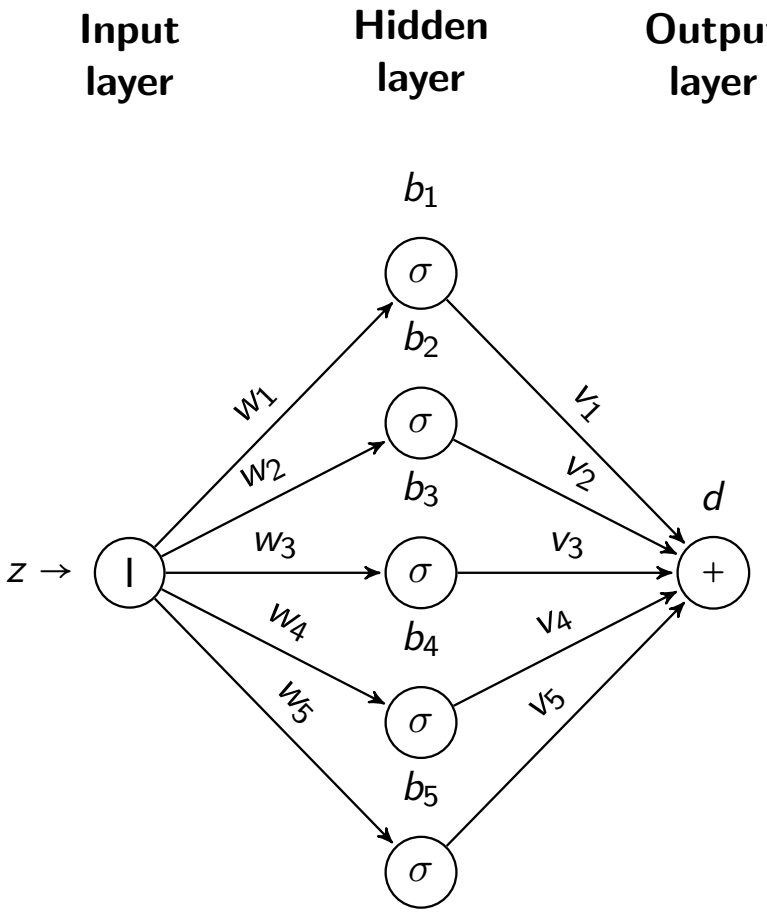
$$K_3 := \frac{q+1}{\eta_1 \lambda_{\min}} \max\{K_1^{1/q}, K_2^{1/q}\}.$$

$k = O(\epsilon^{\frac{q+1}{q}})$ Complexity of standard method is maintained

Part II

1. Use of the multilevel methods for the training of artificial neural networks
2. Application to the solution of PDEs

Artificial neural networks



$z \in \mathbb{R}$,
$p = [v, w, b, d]^T$
$\hat{g}(p, z) \sim g(z)$

Activation funct. σ,
sigmoid: $\sigma(z) = \frac{e^z - 1}{e^z + 1}$,
tanh: $\sigma(z) = \frac{e^{2z} - 1}{e^{2z} + 1}$,
logit: $\sigma(z) = \frac{e^z}{e^z + 1}$, soft-
plus: $\sigma(z) = \log(e^z + 1)$.

Exploit multilevel method for training of ANNs

Training problem:

$$\min_p \mathcal{L}(p, z) = \mathcal{F}(\hat{g}(p, z) - g(z)), \qquad z \in \mathcal{T}$$

$$\hat{g}(p, z) = \sum_{i=1}^{r} v_i \sigma(w_i z + b_i) + d$$

where $\mathcal{L}$ is the loss function, $\mathcal{T}$ training set.

Exploit multilevel method for training of ANNs

Training problem:

$$\min_{p} \mathcal{L}(p, z) = \mathcal{F}(\hat{g}(p, z) - g(z)), \qquad z \in \mathcal{T}$$

$$\hat{g}(p, z) = \sum_{i=1}^{r} v_i \sigma(w_i z + b_i) + d$$

where $\mathcal{L}$ is the loss function, $\mathcal{T}$ training set.

Large-scale problem: can we exploit multilevel methods for the training?

Exploit multilevel method for training of ANNs

Training problem:

$$\min_{p} \mathcal{L}(p, z) = \mathcal{F}(\hat{g}(p, z) - g(z)), \qquad z \in \mathcal{T}$$

$$\hat{g}(p, z) = \sum_{i=1}^{r} v_i \sigma(w_i z + b_i) + d$$

where $\mathcal{L}$ is the loss function, $\mathcal{T}$ training set.

Large-scale problem: can we exploit multilevel methods for the training?

- How to build the coarse problem? The variables to be optimized are the network's weights:
  NO evident geometrical structure to exploit!

Exploit multilevel method for training of ANNs

Training problem:

$$\min_{p} \mathcal{L}(p, z) = \mathcal{F}(\hat{g}(p, z) - g(z)), \qquad z \in \mathcal{T}$$

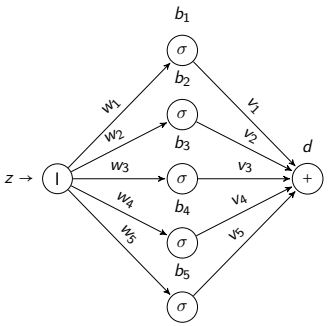
$$\hat{g}(p, z) = \sum_{i=1}^{r} v_i \sigma(w_i z + b_i) + d$$

where $\mathcal{L}$ is the loss function, $\mathcal{T}$ training set.

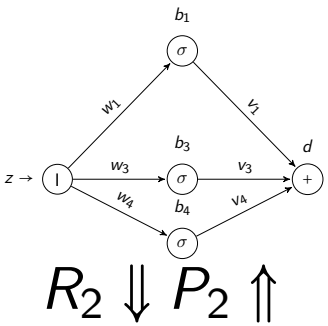
Large-scale problem: can we exploit multilevel methods for the training?

- How to build the coarse problem? The variables to be optimized are the network's weights:
  NO evident geometrical structure to exploit!
- The network possesses a purely algebraic structure: can we exploit it?

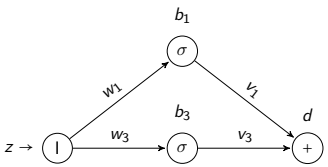
Exploit multilevel method for training of ANNs



$$\mathcal{F}_1 : \mathbb{R}^{3r_1} \to \mathbb{R}$$
$$\hat{g}(p, z) = \sum_{i \in I_1} v_i \sigma(w_i z + b_i) + d$$
$$|I_1| = r_1$$



$$\mathcal{F}_2 : \mathbb{R}^{3r_2} \to \mathbb{R}$$
$$\hat{g}(p, z) = \sum_{i \in I_2} v_i \sigma(w_i z + b_i) + d$$
$$I_2 \subset I_1, \ |I_2| = r_2 < r_1$$



$$\mathcal{F}_3 : \mathbb{R}^{3r_3} \to \mathbb{R}$$
$$\hat{g}(p, z) = \sum_{i \in I_3} v_i \sigma(w_i z + b_i) + d$$
$$I_3 \subset I_2, \ |I_3| = r_3 < r_2$$

How do we select the hierarchy of variables?

Algebraic multigrid: C/F splitting

Ruge and Stueben C/F splitting for $Ax = b$

- Two variables i, j are said to be *coupled* if $a_{i,j} \neq 0$.
- We say that a variable i is strongly coupled to another variable j, if $-a_{i,j} \geq \epsilon \max_{a_{i,k}<0} |a_{i,k}|$ for a fixed $0 < \epsilon < 1$, usually $\epsilon = 0.25$.

Prolongation-Restriction operators

$P = [I; \Delta]$, $R = P^T$.

Which matrix should we use?

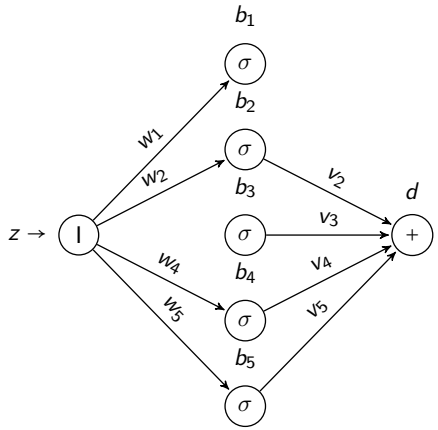
Assume to use a second-order model.
At each iteration we have to solve a linear system of the form:

$$(B_k + \tilde{\lambda}_k I)s = -\nabla f(x_k), \quad \tilde{\lambda}_k > 0.$$

As in AMG for linear systems, we use information contained in matrix B_k.

Which matrix should we use?

Remark
Variables are coupled!
$\{w_i, b_i, v_i\}$



We do not use the full matrix B_k and we define A as:

$$B_k = \begin{bmatrix} f_{v,v} & .. & .. \\ .. & f_{w,w} & .. \\ .. & .. & f_{b,b} \end{bmatrix} \rightarrow A = \frac{f_{v,v}}{\|f_{v,v}\|_\infty} + \frac{f_{w,w}}{\|f_{w,w}\|_\infty} + \frac{f_{b,b}}{\|f_{b,b}\|_\infty}$$

We define the coarse/fine splitting based on the auxiliary matrix A.

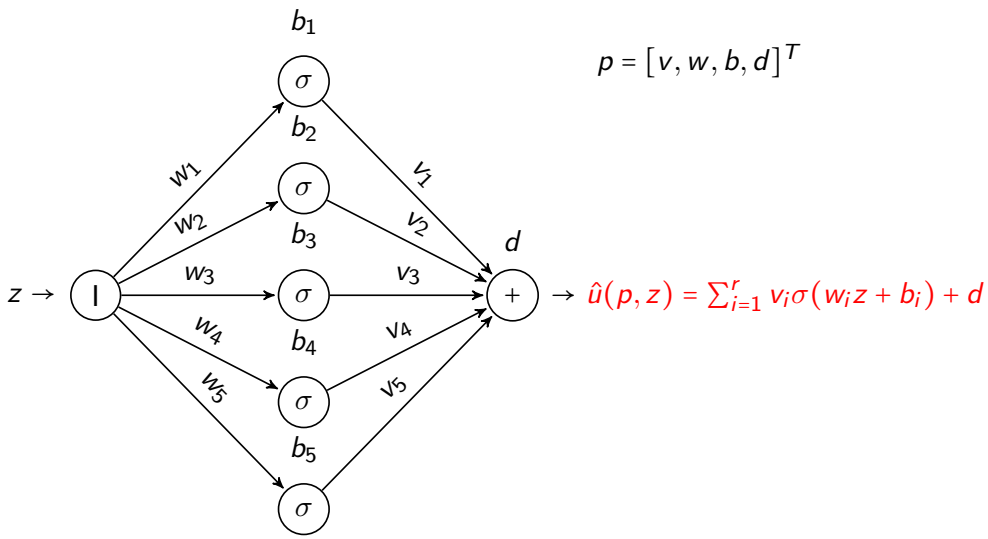
Application: solution of PDEs

Approximate the solution $u(z)$ of a PDE:

$$D(z, u(z)) = g(z), \ z \in (a, b);$$
$$u(a) = A, \ u(b) = B.$$

We approximate the solution of the PDE with a neural network:

$$u(z) \sim \hat{u}(p, z), \quad p \in \mathbb{R}^n$$



Application: solution of PDEs

We select a training set $\mathcal{T}$ s.t. $|\mathcal{T}| = t$:

$$z = [z_1, \ldots, z_t]^T, \quad a \leq z_1 < \cdots < z_t \leq b$$

We define

$$\mathcal{L}(p, z) = \frac{1}{2t}(\|D(z, \hat{u}(z)) - g(z)\|^2 + \lambda_p(\|\hat{u}(a) - A\|^2 + \|\hat{u}(b) - B\|^2))$$

for $\hat{u}(z) \in \mathbb{R}^t$.

Advantages

- No need of discretization: we get an analytical expression of the solution, with good generalization properties (also for points outside the interval)
- We can solve also nonlinear equations, or equations with highly-nonlinear solution

Choice of the true solution

$$D(z, u(z)) = g(z), \ z \in (a, b);$$

- We choose g to have true solution $u_T(z, \nu)$ depending on ν

Remark

- As ν increases the function becomes more oscillatory and it is harder to approximate.
- The size of the problem increases with the number of nodes.
- $\mathcal{T}$: equispaced points in $(0, 1)$ with $h = \frac{1}{3\nu}$ (Shannon's criterion).

Least-squares problem → multi-level Levenberg-Marquardt method

Preliminary results: Poisson's equation 10 runs

1D		$\nu = 20$	$r = 2^9$		$\nu = 25$	$r = 2^{10}$
Solver	iter	RMSE	save	iter	RMSE	save
LM	869	1.e-4		1439	1.e-3	
MLM	507	1.e-4	1.1-2.6-4.3	1325	1.e-3	1.2-1.7-2.8

Table: 1D Poisson's equation, $u_T(z, \nu) = cos(\nu z)$, 10 runs

2D		$\nu = 5$	$r = 2^{10}$		$\nu = 6$	$r = 2^{11}$
Solver	iter	RMSE	save	iter	RMSE	save
LM	633	1.e-3		1213	1.e-3	
MLM	643	1.e-3	1.1-1.5-2.1	1016	1.e-3	1.2-1.9-2.4

Table: 2D Poisson's equation, $u_T(z, \nu) = cos(\nu z)$, 10 runs

save(min,average,max)=ratio between total number of flops required for matrix-vector products

Helmholtz's equation, 10 runs

		$\nu = 5$	$r = 2^{10}$
Solver	`iter`	`RMSE`	`save`
LM	1159	1.e-3	
MLM	1250	1.e-3	1.2-1.9-3.1

Table: Helmholtz's equations. $\Delta u(z) + \nu^2 u(z) = 0$, $u_T(z, \nu) = sin(\nu z) + cos(\nu z)$

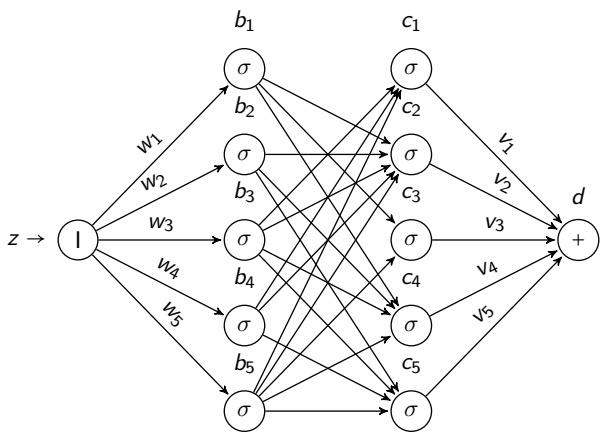
save=ratio between total number of flops required for matrix-vector products

Conclusions

- Theoretical contribution: We have presented a class of multilevel high-order methods for optimization and proved their global convergence and complexity.
- Practical contribution: We have got further insight on the methods proposing a AMG strategy to build coarse representations of the problem to use some methods in the family for the training of artificial neural networks.

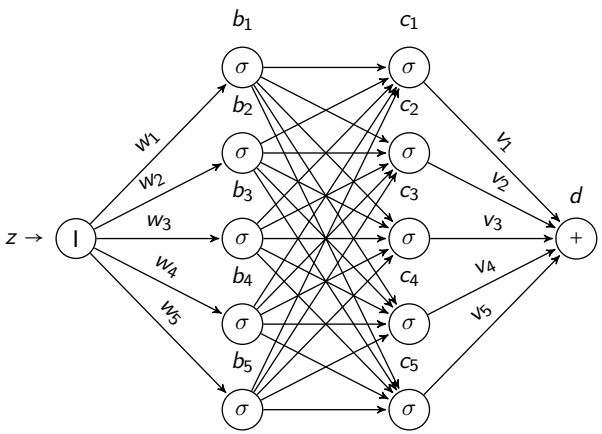
Future work 1: Extend the method to multilayer networks.

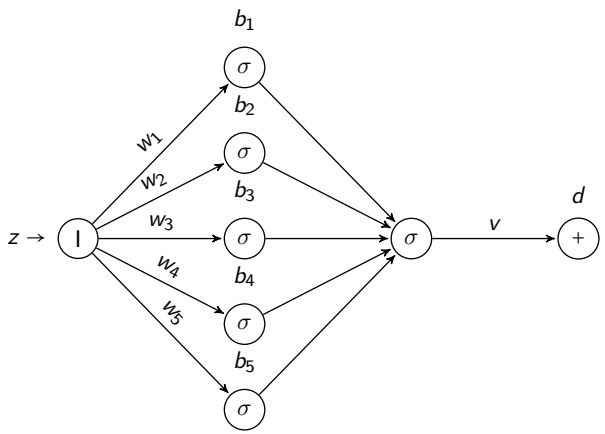
- Extend the method as it is: use a sparse network.



Future work 1: Extend the method to multilayer networks.

- Extend the method as it is: use a sparse network.
- Change strategy to build coarse problems: compress variables in a layer to exploit the structure of the multilayer network.





Future work 2: Hessian-free method

- Make it a competitive training method: method needs to compute and store the Hessian matrix (for step computation and to build transfer operators): too expensive for large-scale problems.
- Hessian complete calculation needed just once (first iteration) to compute R and P.

Thank you for your attention!

For more details:

On high-order multilevel optimization strategies and their application to the training of artificial neural networks

Prolongation operator

$$x_i^h = (Px^H)_i = \begin{cases} x_i^H & \text{if } i \in C, \\ \sum_{k \in P_i} \delta_{i,k} x_k^H & \text{if } i \in F, \end{cases}$$

with

$$\delta_{i,k} = \begin{cases} -\alpha_i a_{i,k}/a_{i,i} & \text{if } k \in P_i^-, \\ -\beta_i a_{i,k}/a_{i,i} & \text{if } k \in P_i^+, \end{cases} \qquad \alpha_i = \frac{\sum_{j \in N_i} a_{i,j}^-}{\sum_{k \in P_i} a_{i,k}^-}, \qquad \beta_i = \frac{\sum_{j \in N_i} a_{i,j}^+}{\sum_{k \in P_i} a_{i,k}^+},$$

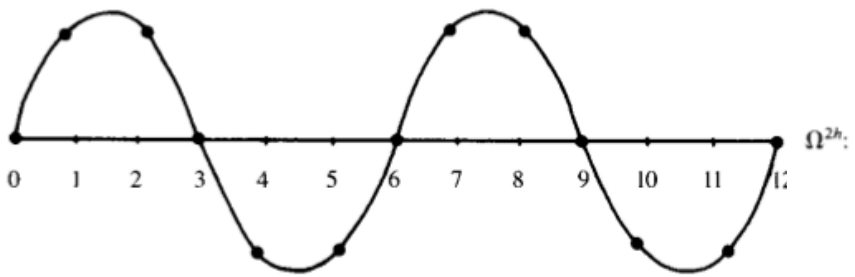
where $a_{i,j}^+ = \max\{a_{i,j}, 0\}$, $a_{i,j}^- = \min\{a_{i,j}, 0\}$, N_i is the set of variables connected to i (i.e. all j such that $a_{i,j} \neq 0$), P_i the set of coarse variables strongly connected to i, which is partitioned in P_i^- (negative couplings) and P_i^+ (positive couplings). The interpolation operator, assuming to have regrouped and ordered the variables to have all those corresponding to indexes in C at the beginning, is then defined as $P = [I; \Delta]$ where I is the identity matrix of size $|C|$ and Δ is the matrix such that $\Delta_{i,j} = \delta_{i,j}$.

Classical multigrid methods

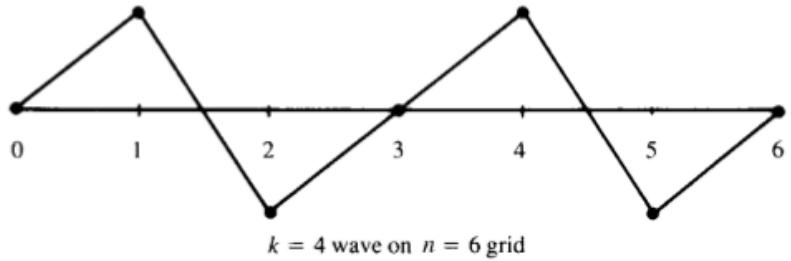
- Consider a linear elliptic PDE: $D(z, u(z)) = f(z)\ z \in \Omega$ + b.c.
- Discretize on grid h. Get a large-scale linear system $A_h x_h = b_h$.

Consider the discretization of the same PDE problem on a coarser grid: $A_H x_H = b_H$, $H > h$.

- Relaxation methods fails to eliminate smooth components of the error efficiently.
- Smooth components projected on a coarser grid appear more oscillatory.



$k = 4$ wave on $n = 12$ grid



$k = 4$ wave on $n = 6$ grid

Coarse problem construction

Define transfer grid operators: P prolongation and R restriction to project vectors from a grid to another: $x_H = Rx_h$, $x_h = Px_H$, such that $R = \alpha P^T$.

Geometry exploitation

The geometrical structure of the problem is exploited to build R and P.

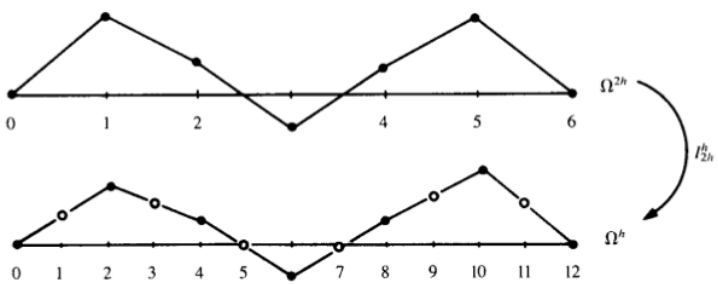


Figure 3.2: *Interpolation of a vector on coarse grid Ω^{2h} to fine grid Ω^{h}.*

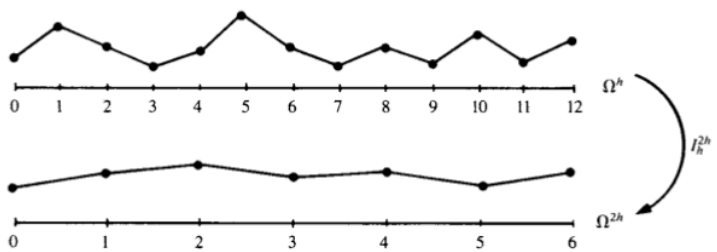


Figure 3.4: *Restriction by full weighting of a fine-grid vector to the coarse grid.*

Basic iterative optimization algorithm

Until convergence

- Define the local model m_k of f around x_k, depending on λ_k
- Compute a trial point $x_k + s_k$ that decreases this model
- Compute the predicted reduction $m_k(x_k) - m_k(x_k + s_k)$
- Evaluate change in the objective function $f(x_k) - f(x_k + s_k)$
- If achieved change $\sim$ predicted reduction then
  - Accept trial point as new iterate $x_{k+1} = x_k + s_k$

  else
  - Reject the trial point $x_{k+1} = x_k$
  - Increase λ_k

Recursive multi-scale q-order methods

Until convergence

- Choose $q \geq 1$. Choose either a Taylor or a (useful) recursive model.
  - Taylor model: compute a Taylor step satisfying a sufficient decrease property
  - Recursive: apply the algorithm recursively
- Evaluate change in the objective function
- If achieved change $\sim$ predicted reduction then
  - Accept trial point as new iterate

  else
  - Reject the trial point
  - Increase λ

The algorithm is proved globally convergent to first order critical points