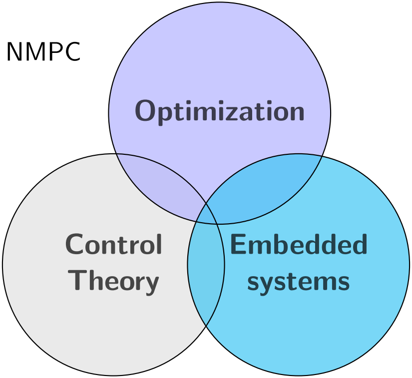
Efficient Numerical Methods for Embedded Optimization in AWE Applications

AWESCO Kickoff Meeting - Andrea Zanelli

System Control and Optimization Laboratory, IMTEK

Few words about me...

- Bachelors in Automation Technologies at Politecnico di Milano
- Masters in Robotics, Systems and Control at ETH Zurich
  - Internship at ABB Corporate Research Center on Embedded Model Predictive Control
  - Master thesis at Embotech on NMPC
  - Research assistant at ETH
- AWESCO fellow at ALUFR

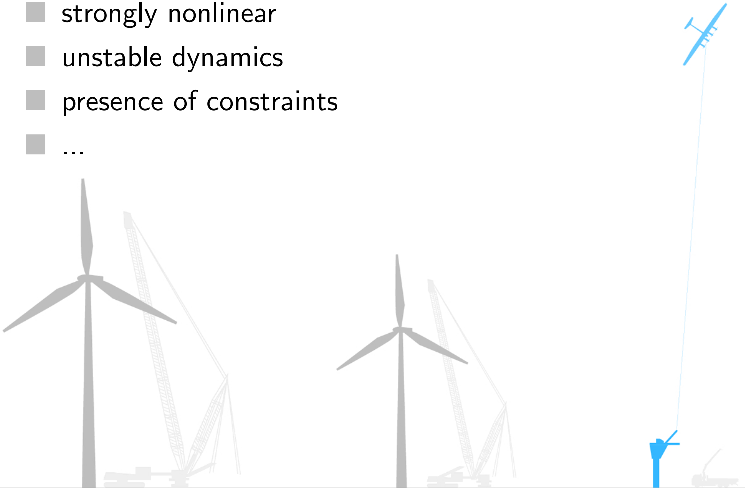


...and my PhD topic

...and my PhD topic

Challenging control problems:

- strongly nonlinear
- unstable dynamics
- presence of constraints
- ...



...and my PhD topic

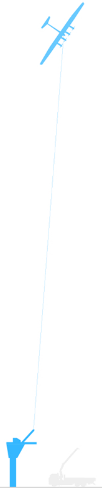
Challenging control problems:

- strongly nonlinear
- unstable dynamics
- presence of constraints
- ...

→ need for advanced control techniques

→ optimization-based solutions

→ several open questions and challenges



...and my PhD topic

Challenging control problems:

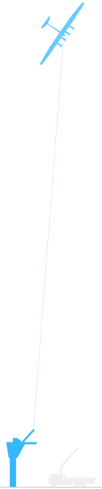
- strongly nonlinear
- unstable dynamics
- presence of constraints
- ...

→ need for advanced control techniques

→ optimization-based solutions

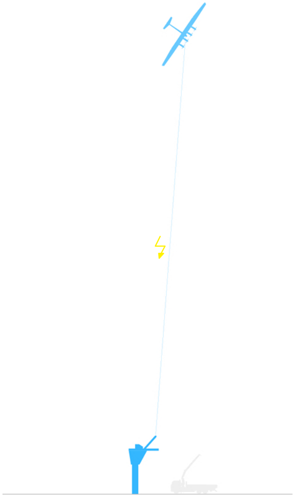
→ several open questions and challenges

main focus: **efficient numerical methods for embedded optimization**



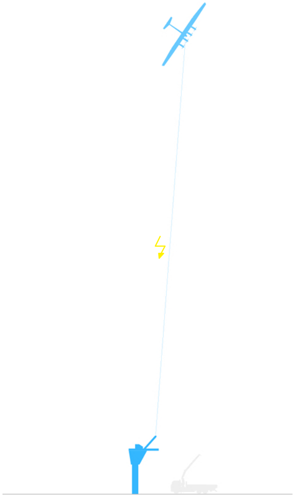
Efficient Numerical Methods for Embedded Optimization

$$\max_{x} \quad f(x)$$



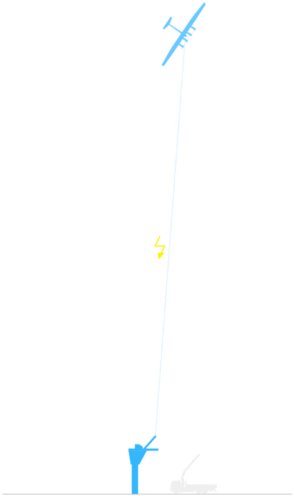
Efficient Numerical Methods for Embedded Optimization

$$\begin{aligned} \max_{x} \quad & f(x) \\ s.t. \quad & g(x) \quad = 0 \end{aligned}$$



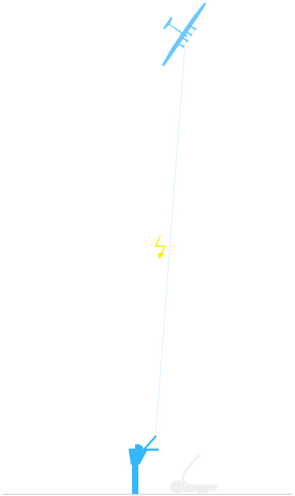
Efficient Numerical Methods for Embedded Optimization

$$
\begin{aligned}
\max_{x} \quad & f(x) \\
s.t. \quad & g(x) = 0 \\
& h(x) \leq 0
\end{aligned}
$$



Efficient Numerical Methods for Embedded Optimization

$$\begin{aligned} \max_{x} \quad & f(x) & & \\ s.t. \quad & g(x) & = 0 & \\ & h(x) & \leq 0 & \end{aligned} \quad \longrightarrow$$

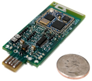


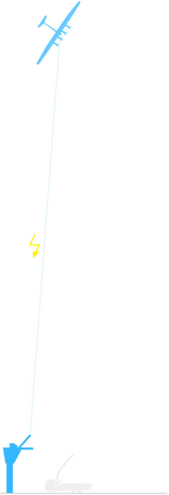
Efficient Numerical Methods for Embedded Optimization

$$
\begin{aligned}
\max_{x} \quad & f(x) \\
s.t. \quad & g(x) = 0 \\
& h(x) \leq 0
\end{aligned}
$$

$\longrightarrow$

$\longrightarrow$

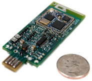


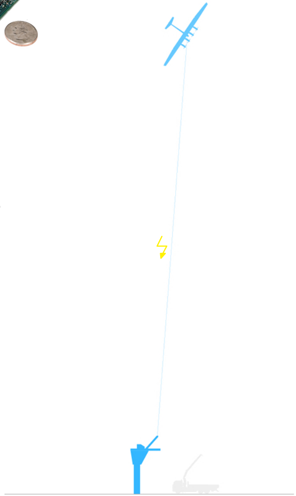


Efficient Numerical Methods for Embedded Optimization

$$\begin{aligned} \max_{x} \quad & f(x) \\ s.t. \quad & g(x) &= 0 \\ & h(x) &\leq 0 \end{aligned} \longrightarrow$$

$\longrightarrow$



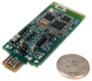


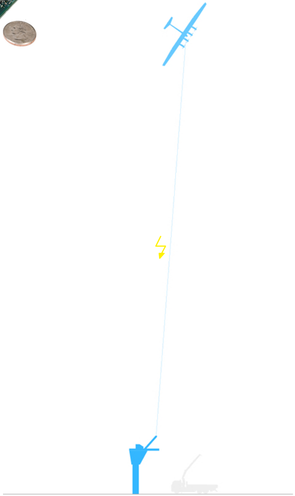
- integrate **smart decision making** into **embedded systems**

Efficient Numerical Methods for Embedded Optimization

$$\begin{aligned} \max_{x} \quad & f(x) \\ s.t. \quad & g(x) = 0 \\ & h(x) \leq 0 \end{aligned} \longrightarrow$$

$\longrightarrow$



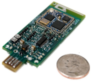


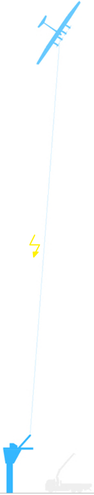
- integrate **smart decision making** into **embedded systems**
- challenges:
  - efficiency at **algorithmic level**

Efficient Numerical Methods for Embedded Optimization

$$\begin{aligned} \max_{x} \quad & f(x) \\ s.t. \quad & g(x) = 0 \\ & h(x) \leq 0 \end{aligned} \longrightarrow$$

$\longrightarrow$





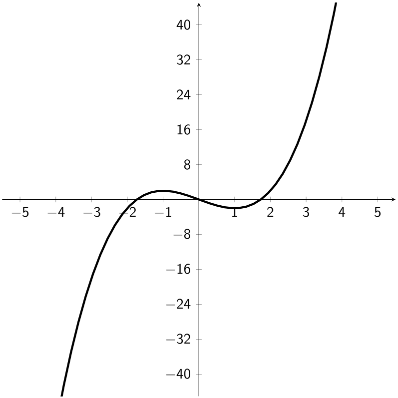
- integrate **smart decision making** into **embedded systems**
- challenges:
  - efficiency at **algorithmic level**
  - efficiency at **implementation level**

Sequential Quadratic Programming

<u>Problem formulation:</u>

$$\min_{\substack{x_0,\cdots,x_N\\u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i, u_i) \\ & x_N = 0 \end{aligned}$$



<u>Iterative approximation:</u>

$$\min_{\substack{x_0,\cdots,x_N\\u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

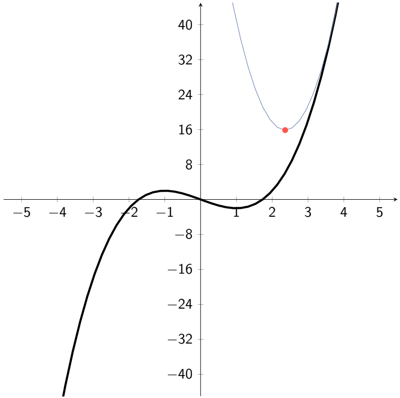
$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i^k, u_i^k) + A_i^k(x_i - x_i^k) + B_i^k(u_i - u_i^k) \\ & x_N = 0 \end{aligned}$$

Sequential Quadratic Programming

Problem formulation:

$$\min_{\substack{x_0,\cdots,x_N \\ u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i, u_i) \\ & x_N = 0 \end{aligned}$$



Iterative approximation:

$$\min_{\substack{x_0,\cdots,x_N \\ u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

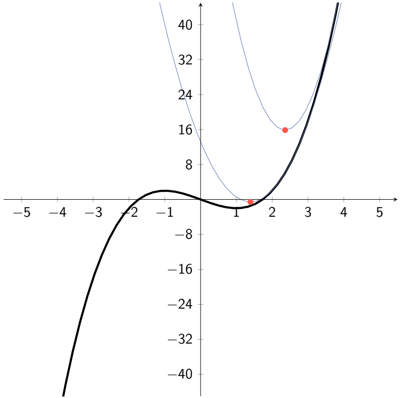
$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i^k, u_i^k) + A_i^k(x_i - x_i^k) + B_i^k(u_i - u_i^k) \\ & x_N = 0 \end{aligned}$$

Sequential Quadratic Programming

Problem formulation:

$$\min_{\substack{x_0,\cdots,x_N\\u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i, u_i) \\ & x_N = 0 \end{aligned}$$



Iterative approximation:

$$\min_{\substack{x_0,\cdots,x_N\\u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

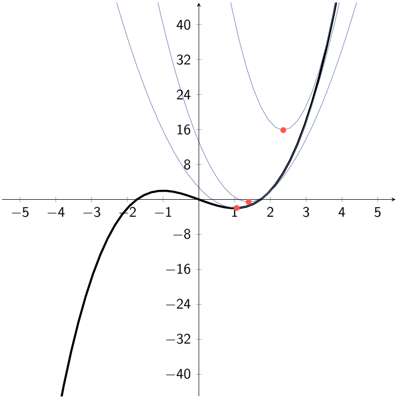
$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i^k, u_i^k) + A_i^k(x_i - x_i^k) + B_i^k(u_i - u_i^k) \\ & x_N = 0 \end{aligned}$$

Sequential Quadratic Programming

Problem formulation:

$$\min_{\substack{x_0,\cdots,x_N\\u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i, u_i) \\ & x_N = 0 \end{aligned}$$



Iterative approximation:

$$\min_{\substack{x_0,\cdots,x_N\\u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

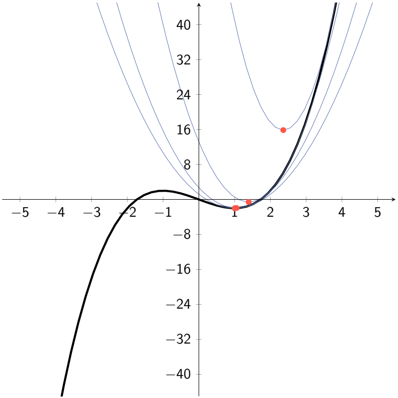
$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i^k, u_i^k) + A_i^k(x_i - x_i^k) + B_i^k(u_i - u_i^k) \\ & x_N = 0 \end{aligned}$$

Sequential Quadratic Programming

Problem formulation:

$$\min_{\substack{x_0,\cdots,x_N \\ u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i, u_i) \\ & x_N = 0 \end{aligned}$$



Iterative approximation:

$$\min_{\substack{x_0,\cdots,x_N \\ u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i^k, u_i^k) + A_i^k(x_i - x_i^k) + B_i^k(u_i - u_i^k) \\ & x_N = 0 \end{aligned}$$

An Efficient Inexact Scheme

$$
\begin{aligned}
\min_{\substack{x_0,\cdots,x_N \\ u_0,\cdots,u_{N-1}}} \quad & \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right) \\
s.t. \quad & x_0 - \bar{x}_0 = 0 \\
& x_{i+1} = f(x_i^k, u_i^k) + A_i^k(x_i - x_i^k) + B_i^k(u_i - u_i^k) \\
& x_N = 0
\end{aligned}
$$

An Efficient Inexact Scheme

$$
\begin{aligned}
\min_{\substack{x_0,\cdots,x_N\\u_0,\cdots,u_{N-1}}} \quad & \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right) \\
s.t. \quad & x_0 - \bar{x}_0 = 0 \\
& x_{i+1} = f(x_i^k, u_i^k) + A_i^k(x_i - x_i^k) + B_i^k(u_i - u_i^k) \\
& x_N = 0
\end{aligned}
$$

$$
\boxed{A_i^k = A = \frac{\partial f}{\partial x_i}(0,0) \quad B_i^k = B = \frac{\partial f}{\partial u_i}(0,0)}
$$

An Efficient Inexact Scheme

$$\min_{\substack{x_0,\cdots,x_N \\ u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

$$\begin{aligned} s.t. \quad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i^k, u_i^k) + A_i^k(x_i - x_i^k) + B_i^k(u_i - u_i^k) \\ & x_N = 0 \end{aligned}$$

$$\boxed{A_i^k = A = \frac{\partial f}{\partial x_i}(0,0) \quad B_i^k = B = \frac{\partial f}{\partial u_i}(0,0)}$$

- no sensitivity generation
- offline condensing

An Efficient Inexact Scheme

$$\min_{\substack{x_0,\cdots,x_N\\u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

$$\begin{aligned}
s.t. \qquad & x_0 - \bar{x}_0 = 0 \\
& x_{i+1} = f(x_i^k, u_i^k) + A_i^k(x_i - x_i^k) + B_i^k(u_i - u_i^k) \\
& x_N = 0
\end{aligned}$$

$$\boxed{A_i^k = A = \frac{\partial f}{\partial x_i}(0,0) \quad B_i^k = B = \frac{\partial f}{\partial u_i}(0,0)}$$

- no sensitivity generation
- offline condensing

$\rightarrow$ only QP solve and forward simulation [Bock et al, 2007]

An Efficient Inexact Scheme

$$\min_{\substack{x_0,\cdots,x_N \\ u_0,\cdots,u_{N-1}}} \quad \frac{1}{2}\sum_{i=0}^{N-1}\left(x_i^T Q x_i + u_i^T R u_i\right)$$

$$\begin{aligned} s.t. \qquad & x_0 - \bar{x}_0 = 0 \\ & x_{i+1} = f(x_i^{k}, u_i^{k}) + A_i^{k}(x_i - x_i^{k}) + B_i^{k}(u_i - u_i^{k}) \\ & x_N = 0 \end{aligned}$$

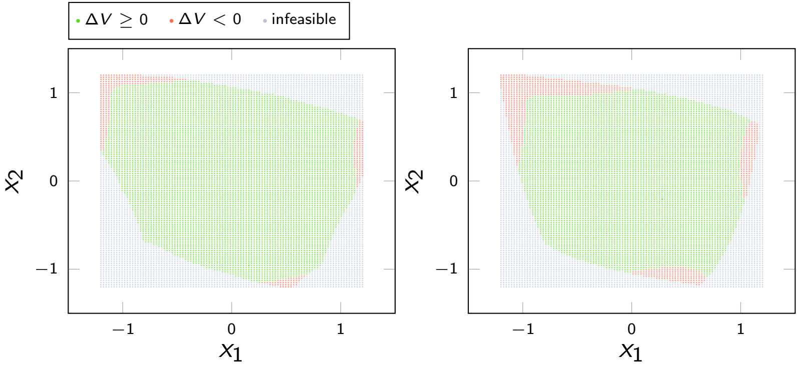
$$\boxed{A_i^{k} = A = \frac{\partial f}{\partial x_i}(0,0) \quad B_i^{k} = B = \frac{\partial f}{\partial u_i}(0,0)}$$

- no sensitivity generation
- offline condensing

$\rightarrow$ only QP solve and forward simulation [Bock et al, 2007]

$\rightarrow$ **how is stability affected?**

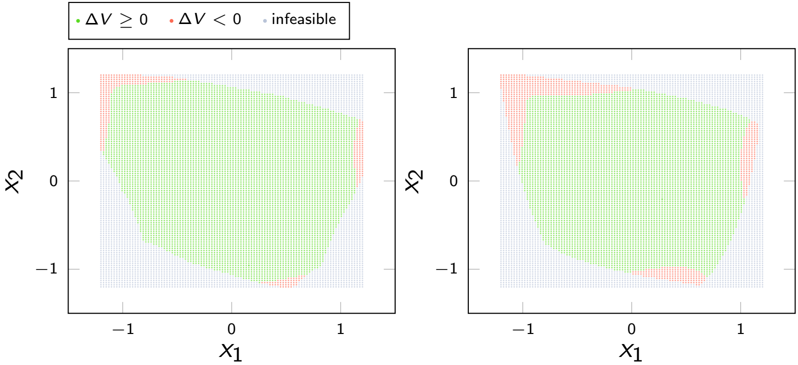
Result: stability can be guaranteed



→ **stability preserved** [Zanelli, Quirynen, Diehl - 2016 (submitted)]

→ **feasibility guaranteed**

Result: stability can be guaranteed



→ **stability preserved** [Zanelli, Quirynen, Diehl - 2016 (submitted)]

→ **feasibility guaranteed**

→ **computational burden** reduced up to 70%

Efficient Linear Algebra for Embedded Optimization

- main operations boil down to **linear algebra** routines:
  - matrix multiplications
  - matrix factorizations
  - ...

Efficient Linear Algebra for Embedded Optimization

- main operations boil down to **linear algebra** routines:
  - matrix multiplications
  - matrix factorizations
  - ...
- main **computational bottleneck**

Efficient Linear Algebra for Embedded Optimization

- main operations boil down to **linear algebra** routines:
  - matrix multiplications
  - matrix factorizations
  - ...

- main **computational bottleneck**

- how to improve efficiency?
  - reduce algorithm complexity
  - optimize **code efficiency**

Efficient Linear Algebra for Embedded Optimization

- main operations boil down to **linear algebra** routines:
  - matrix multiplications
  - matrix factorizations
  - ...
- main **computational bottleneck**
- how to improve efficiency?
  - reduce algorithm complexity
  - optimize **code efficiency**

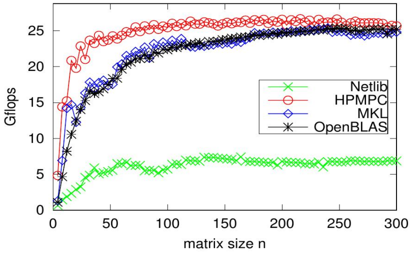
- modern CPUs have complex architectures:
  - caching effects
  - vectorized instruction
  - highly pipelined

Efficient Linear Algebra for Embedded Optimization

- main operations boil down to **linear algebra** routines:
  - matrix multiplications
  - matrix factorizations
  - ...
- main **computational bottleneck**
- how to improve efficiency?
  - reduce algorithm complexity
  - optimize **code efficiency**
- modern CPUs have complex architectures:
  - caching effects
  - vectorized instruction
  - highly pipelined
- how much do we gain?

Efficient Linear Algebra for Embedded Optimization

- main operations boil down to **linear algebra** routines:
  - matrix multiplications
  - matrix factorizations
  - ...
- main **computational bottleneck**
- how to improve efficiency?
  - reduce algorithm complexity
  - optimize **code efficiency**
- modern CPUs have complex architectures:
  - caching effects
  - vectorized instruction
  - highly pipelined
- how much do we gain? **a lot!** [Frison, 2013]



Scope and Project Plan

Objectives:

- tackle challenging control problems in AWE applications
- development of novel **numerical methods**
- **efficiency** at both algorithmic and implementation level

Secondments:

- Chalmers - 2 months
- Makani - 2 months
- ETH Zurich - 2 months