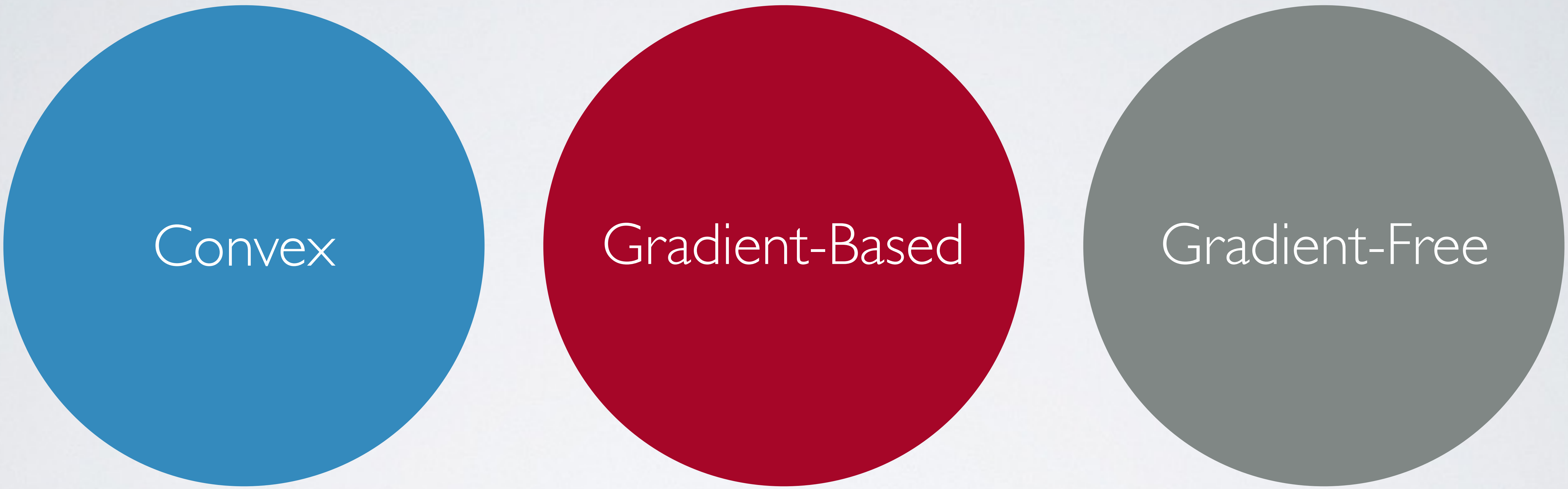
Wind Turbine Optimization

a case study to help you think about your project

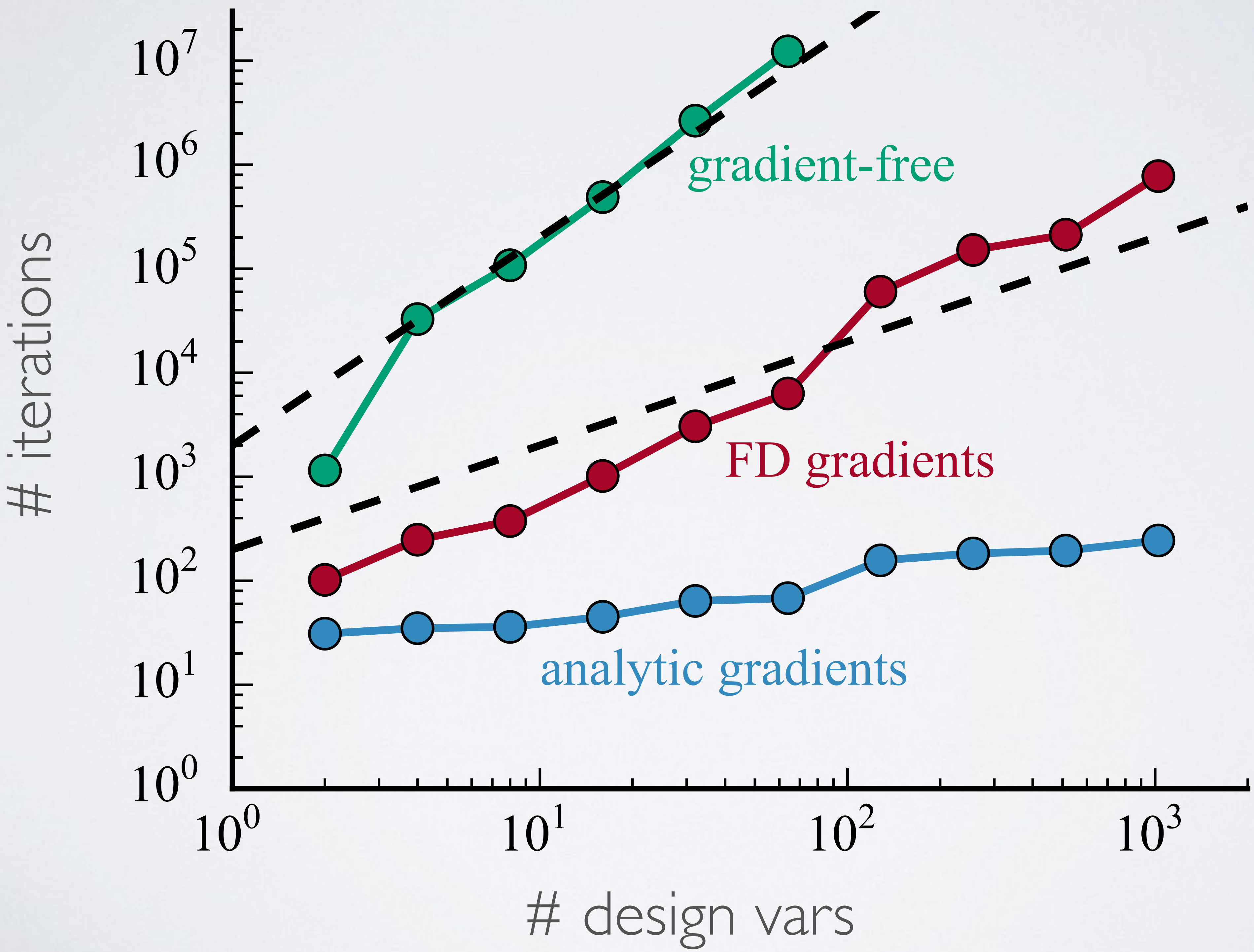
Andrew Ning
ME 575

Three broad areas of optimization (with some overlap)

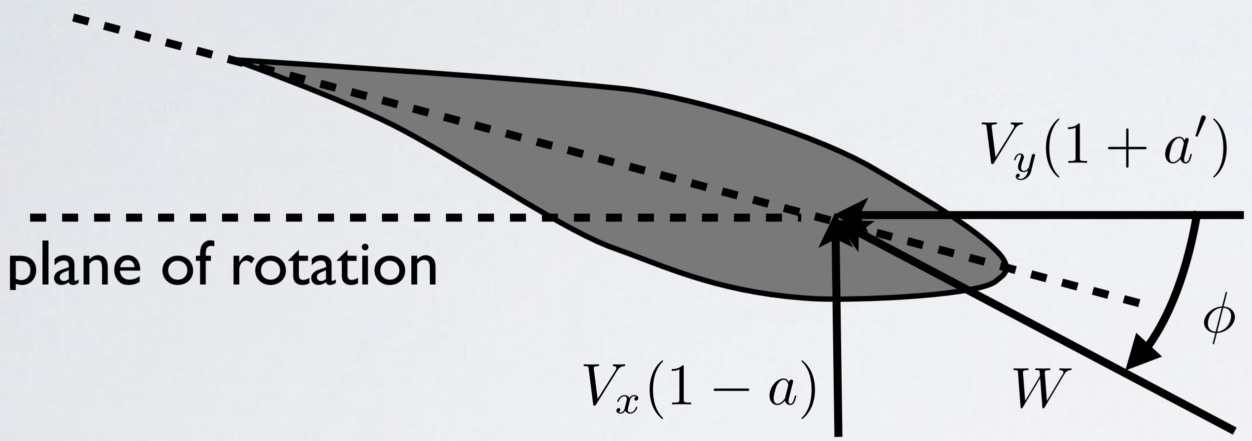


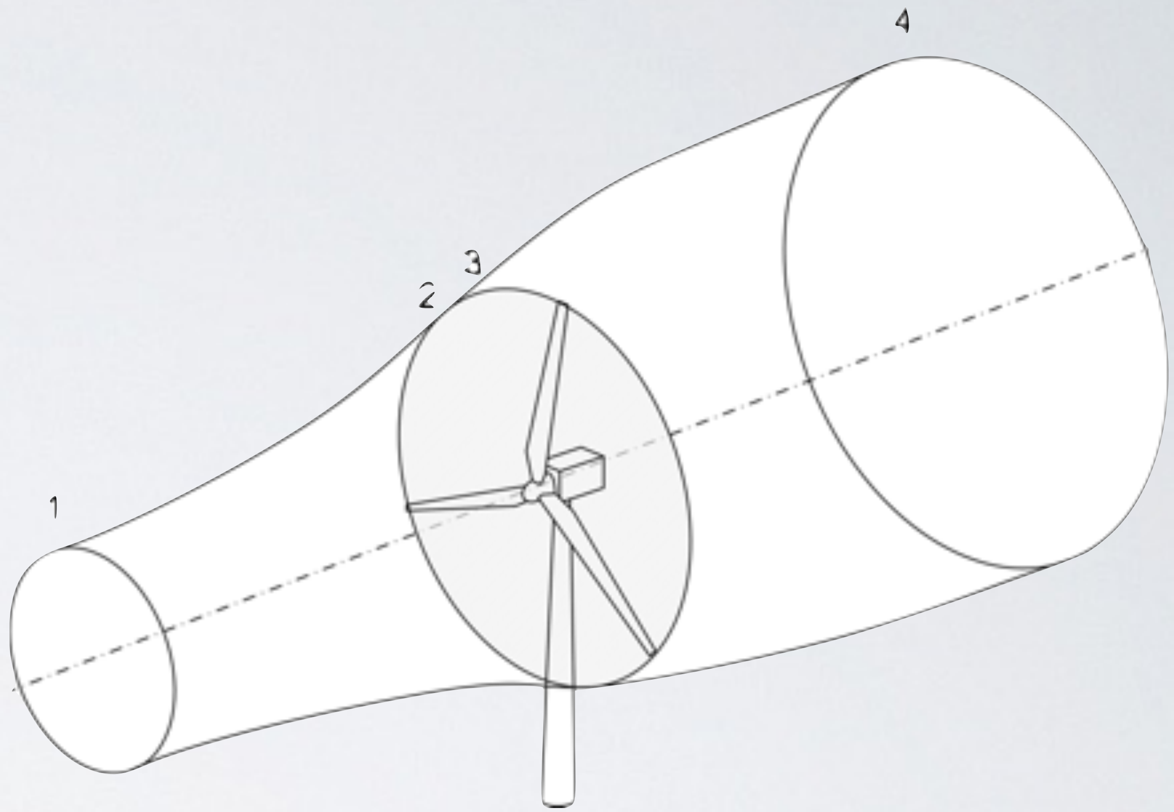
Today we will focus on gradient-based, but for some projects the other areas are more important.

In higher dimensional space, analytic gradients become increasingly important

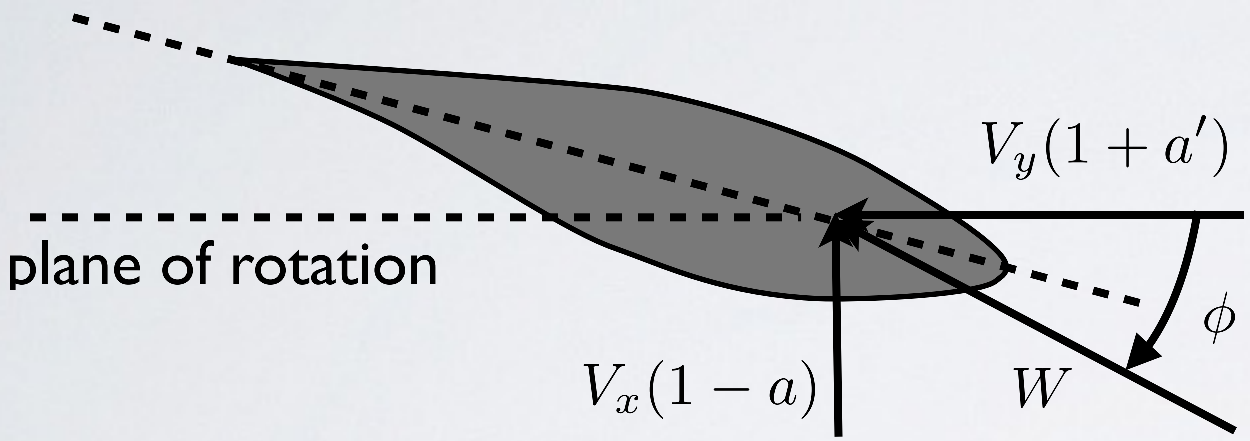


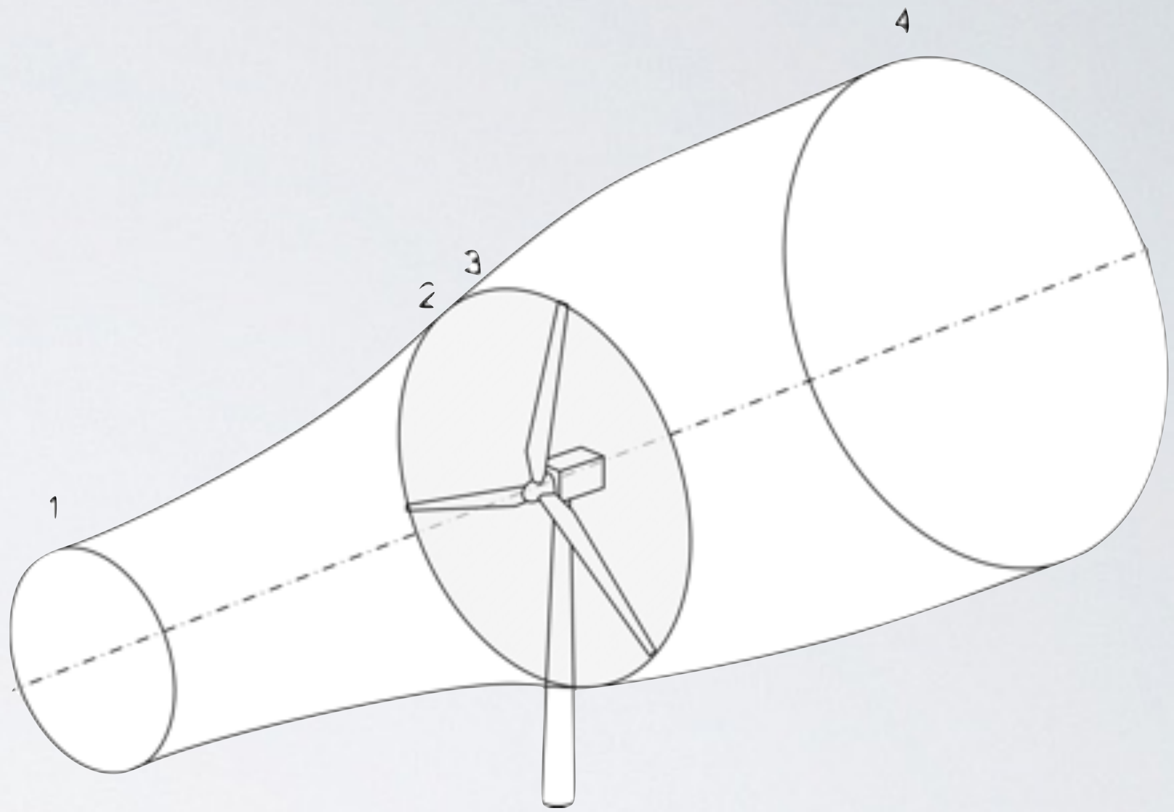
There is a difference between developing for analysis vs optimization





There is a difference between developing for analysis vs optimization





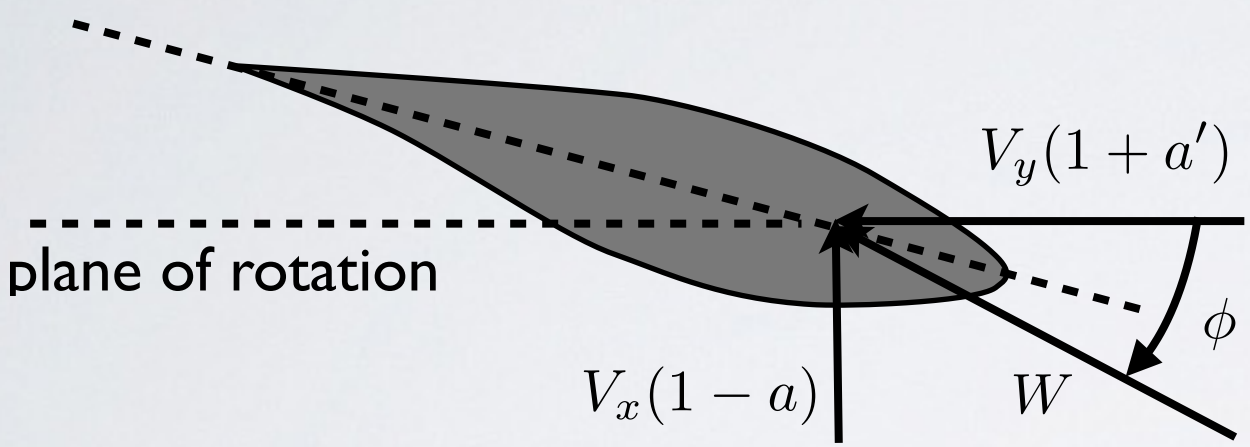
$$C_T = \left(\frac{1-a}{\sin\phi}\right)^2 c_n \sigma'$$

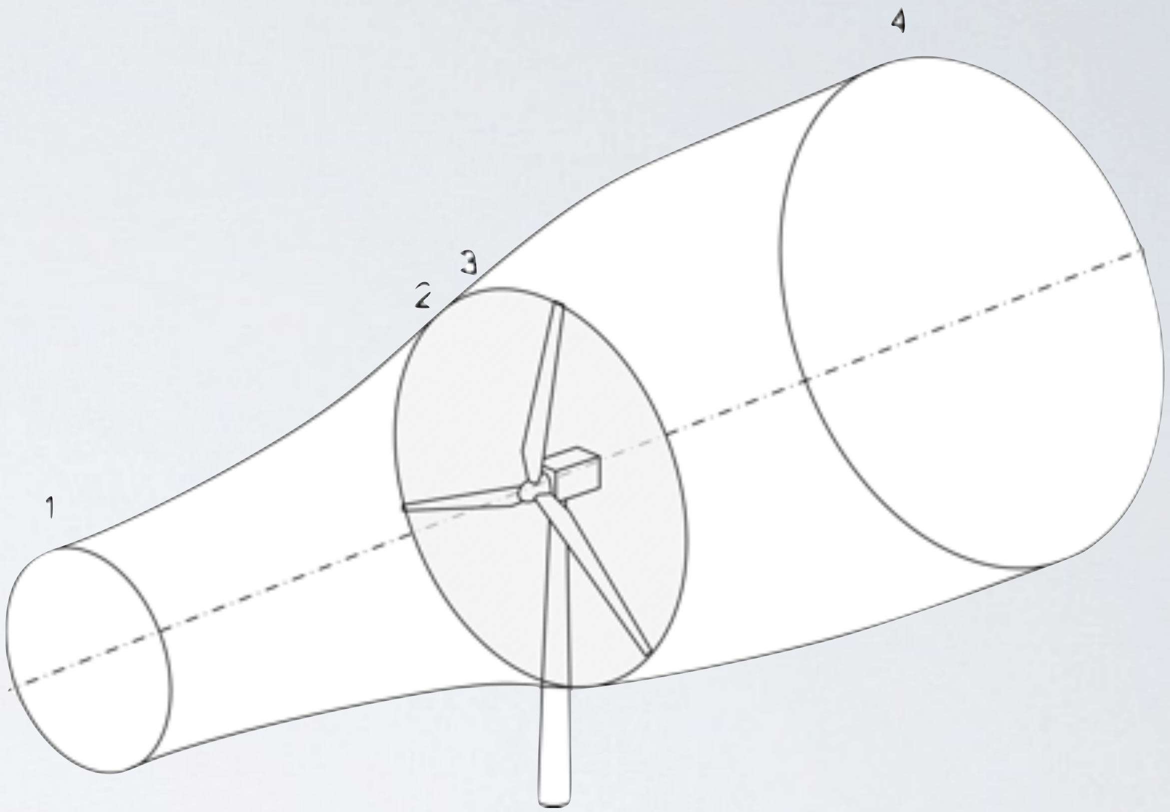
$$C_T = 4a(1-a)$$

$$C_Q = \left(\frac{1+a'}{\cos\phi}\right)^2 c_t \sigma' \lambda_r^2$$

$$C_Q = 4a'(1+a')\tan\phi \lambda_r^2$$

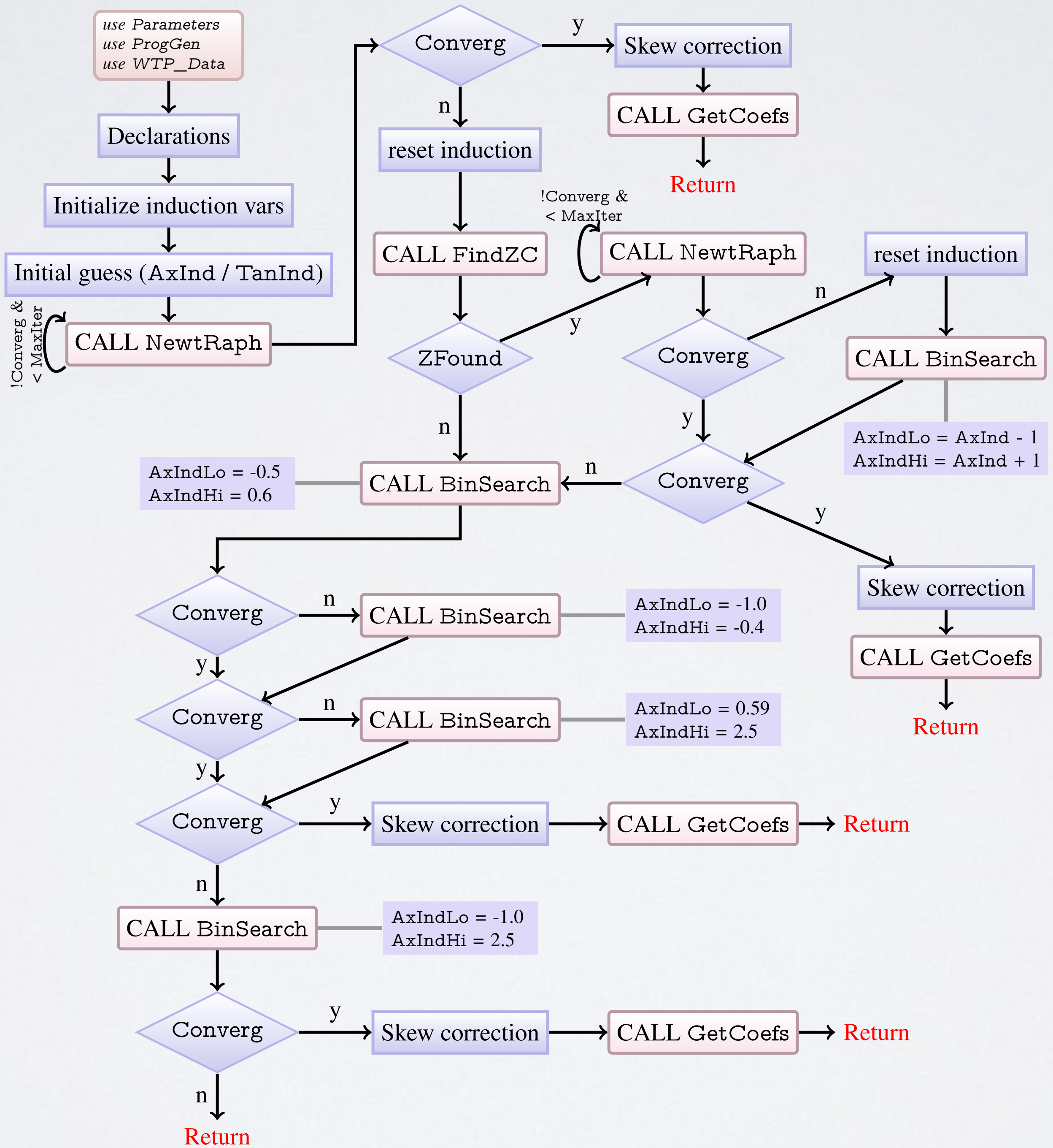
There is a difference between developing for analysis vs optimization



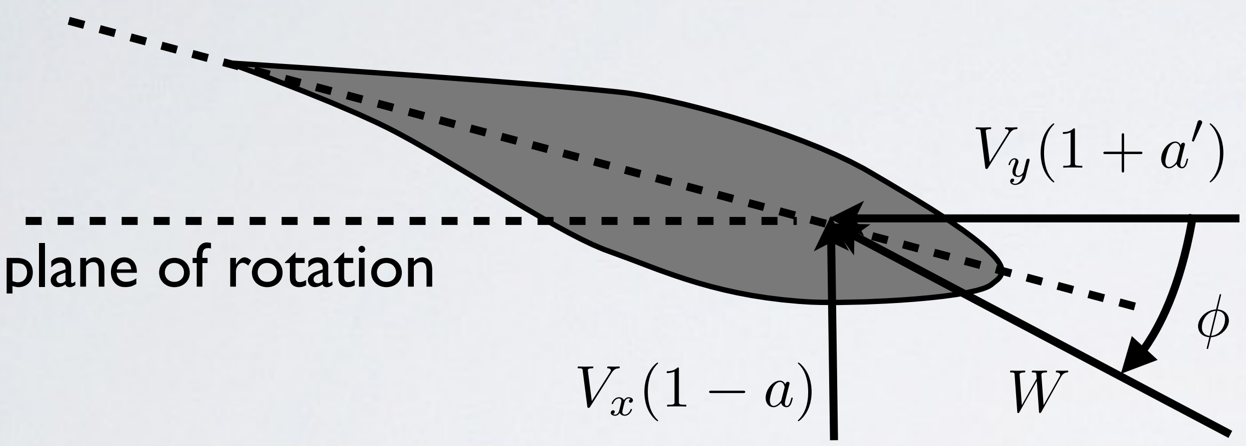


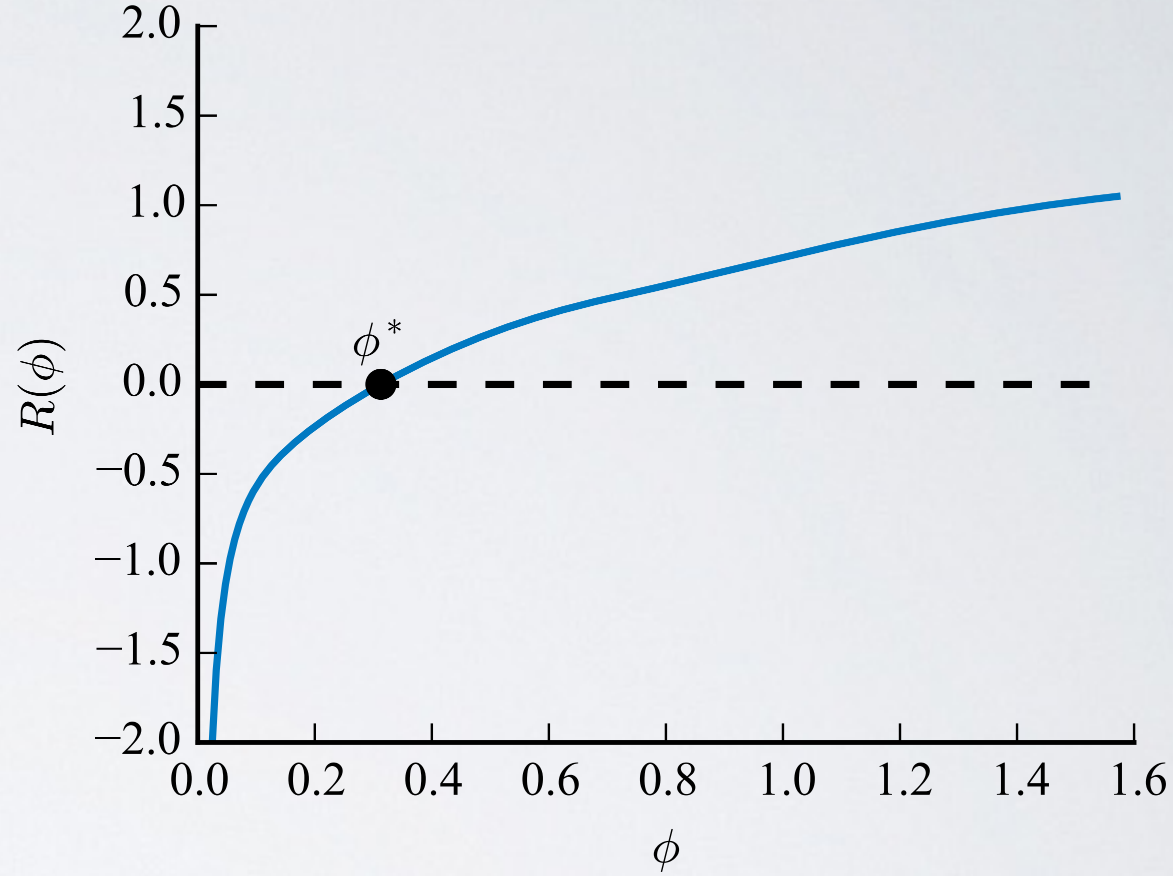
$$a = \frac{1}{\frac{4\sin^2\phi}{c_n\sigma'} + 1} \qquad a' = \frac{1}{\frac{4\sin\phi\cos\phi}{c_t\sigma'} - 1}$$

There is a difference between developing for analysis vs optimization



Sometimes a new solution approach is needed…



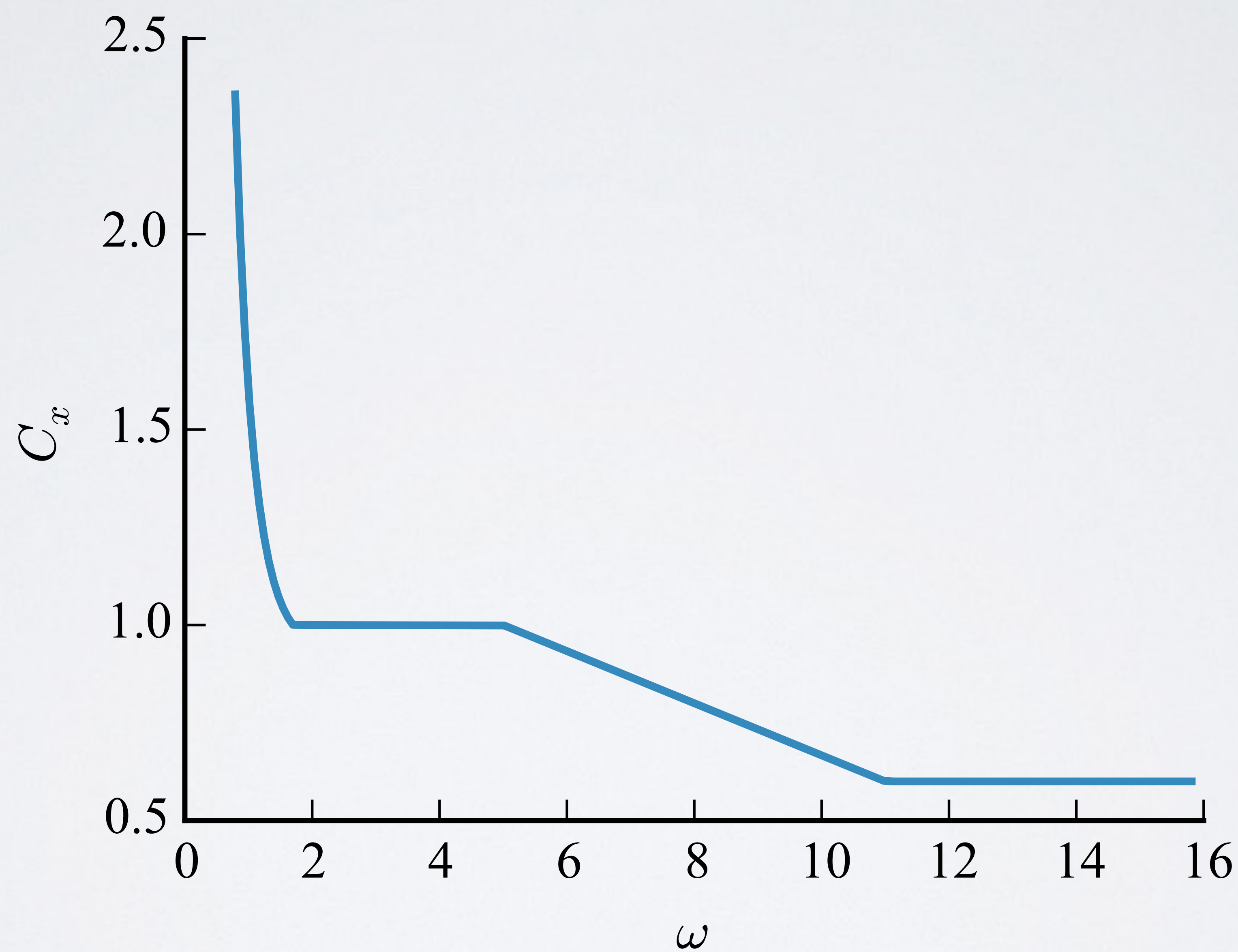


$$\mathcal{R}(\phi) = \frac{\sin\phi}{1-a} - \frac{V_x}{V_y}\frac{\cos\phi}{(1+a')} = 0$$

Sometimes a new solution approach is needed…

Algorithm	Avg. Function Calls	Failure Rate (%)
Steffensen	16.4	16.3
Powell Hybrid	72.3	16.2
Fixed-Point	31.8	12.6
Levenberg-Marquardt	92.3	8.8
Newton	79.0	5.8
New Method	11.3	0.0

But most of the time the changes needed are relatively minor



smoothing empirical factors/corrections

But most of the time the changes needed are relatively minor

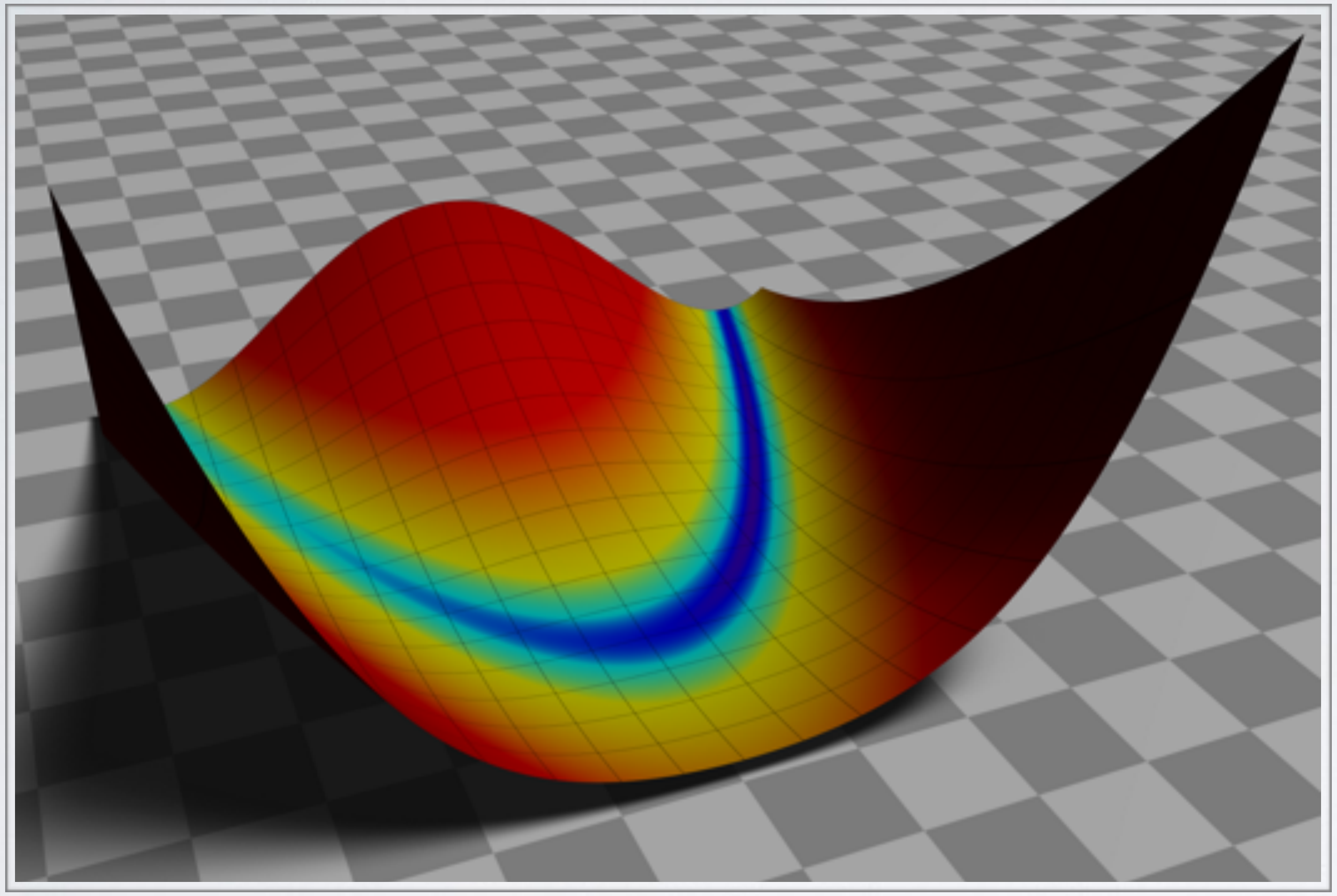
provide alternatives to input files

Planning for optimization at the beginning can help a lot

Try to avoid:

- max/min
- abs
- piecewise functions
- convergence loops,
- noisy output
- empirical models
- discretization

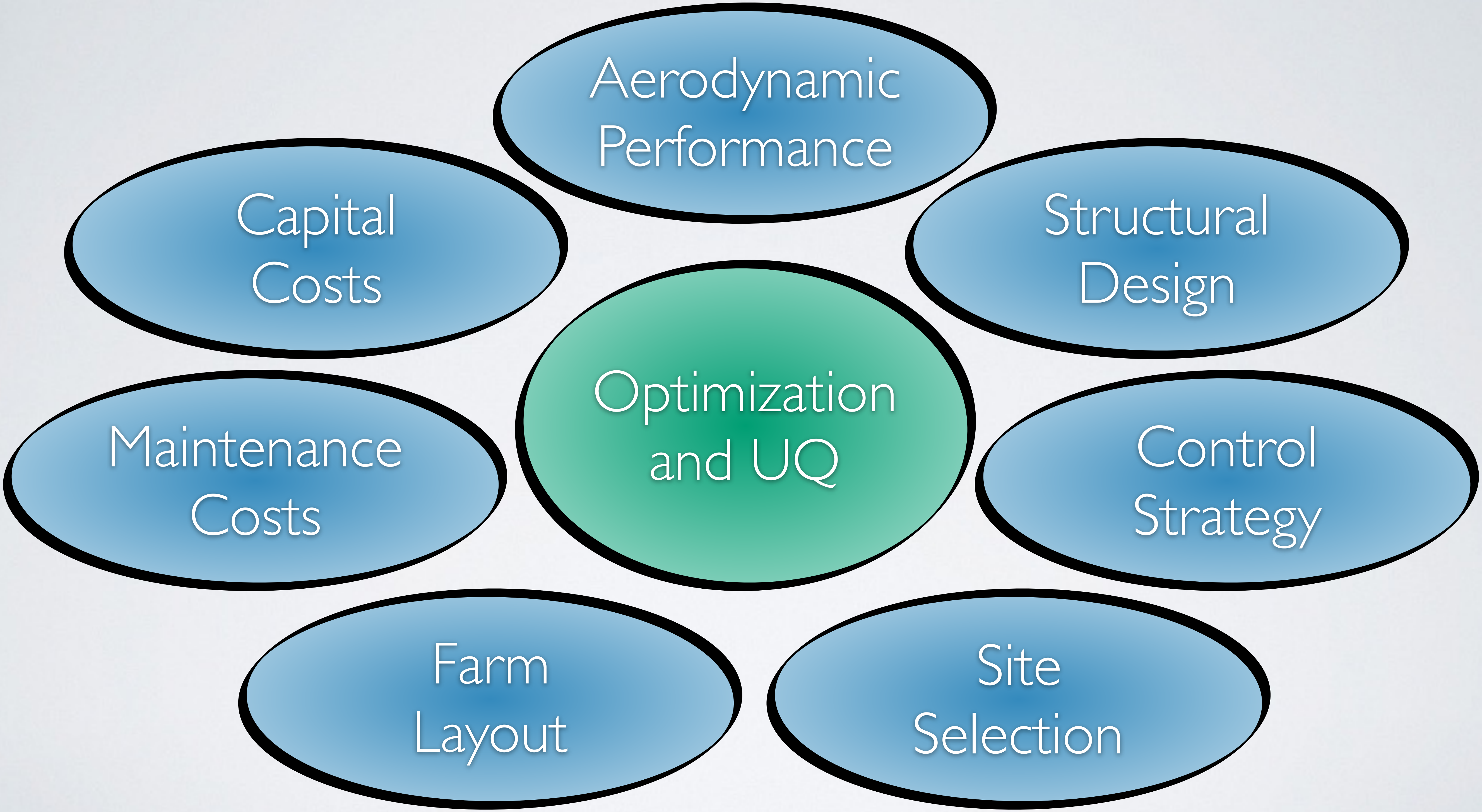
Planning for optimization at the beginning can help a lot



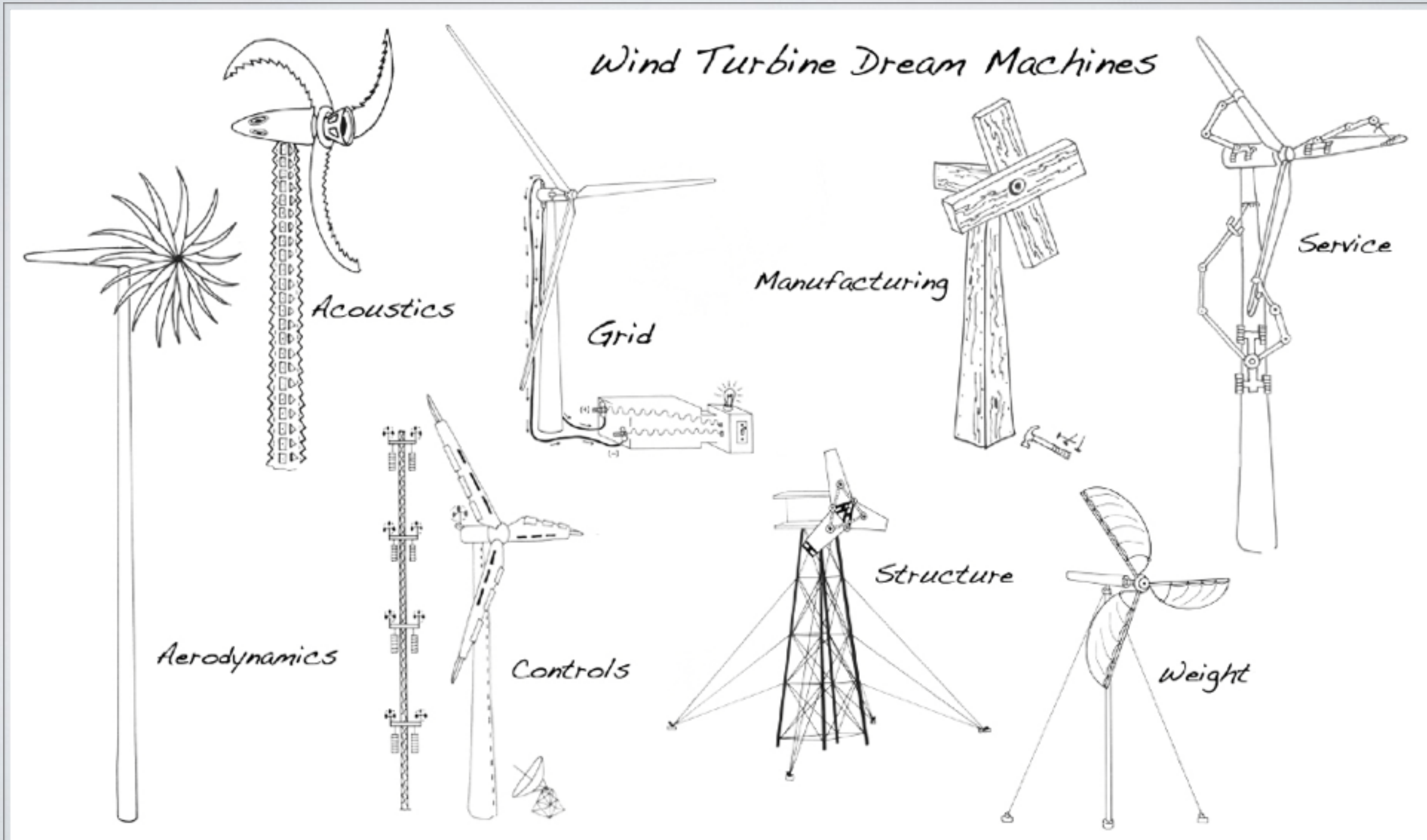
[f, g] = func(x)

Think about gradients upfront

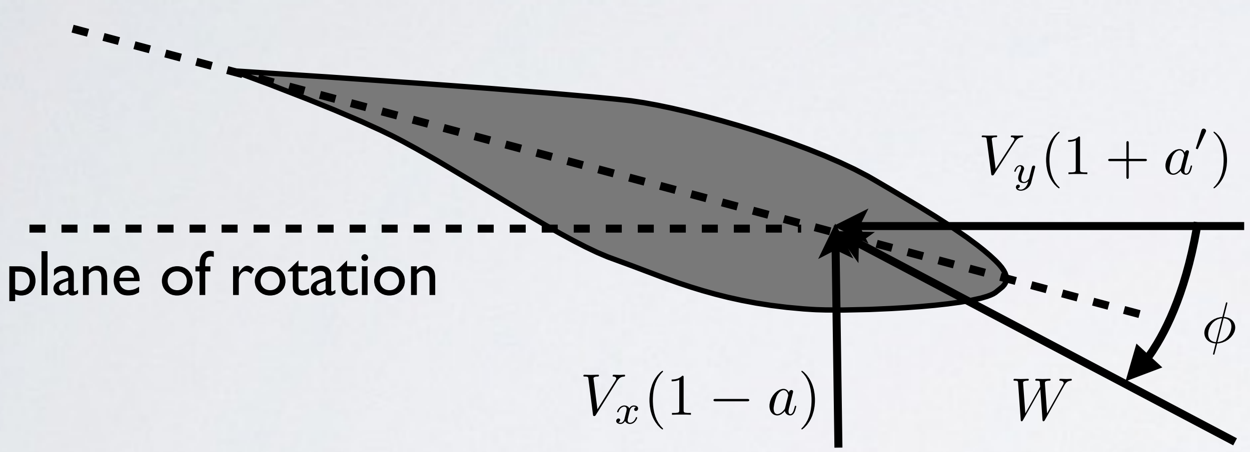
Engineering design is multidisciplinary



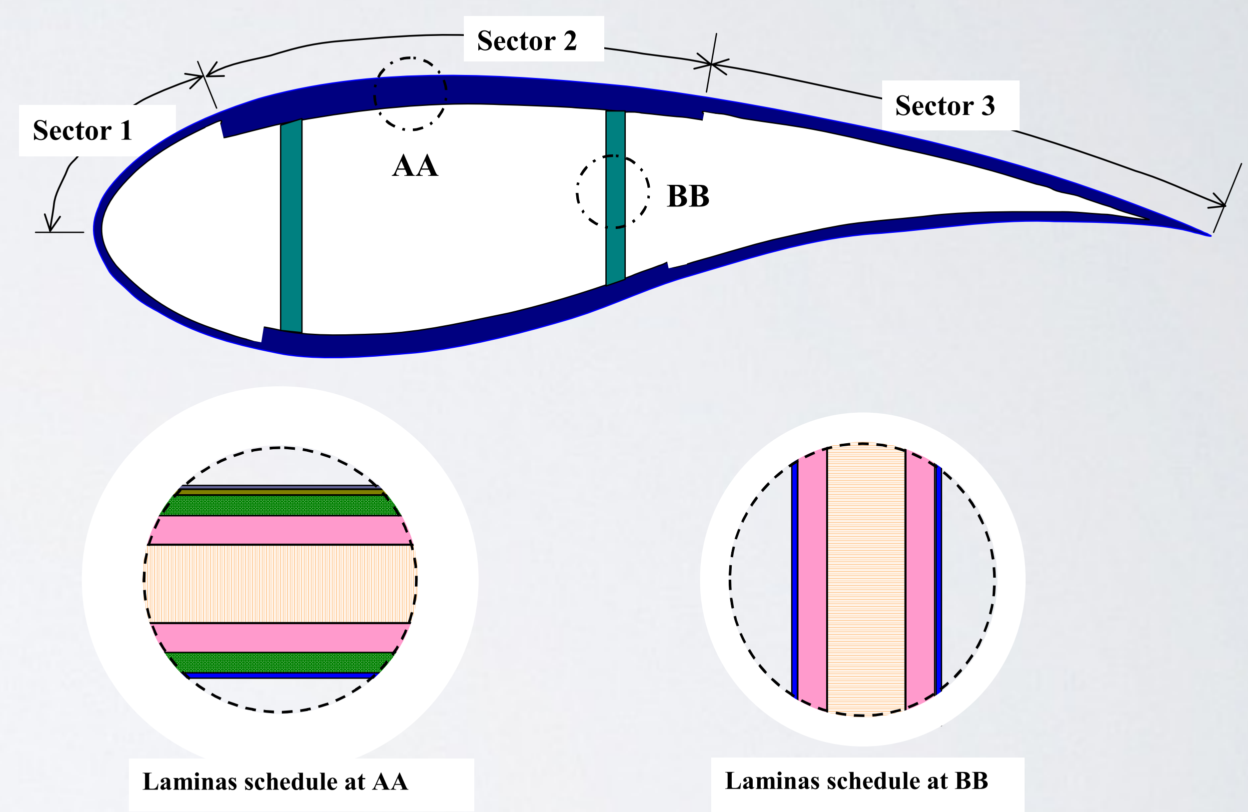
Single discipline thinking usually leads to poor solutions



Single discipline thinking usually leads to poor solutions

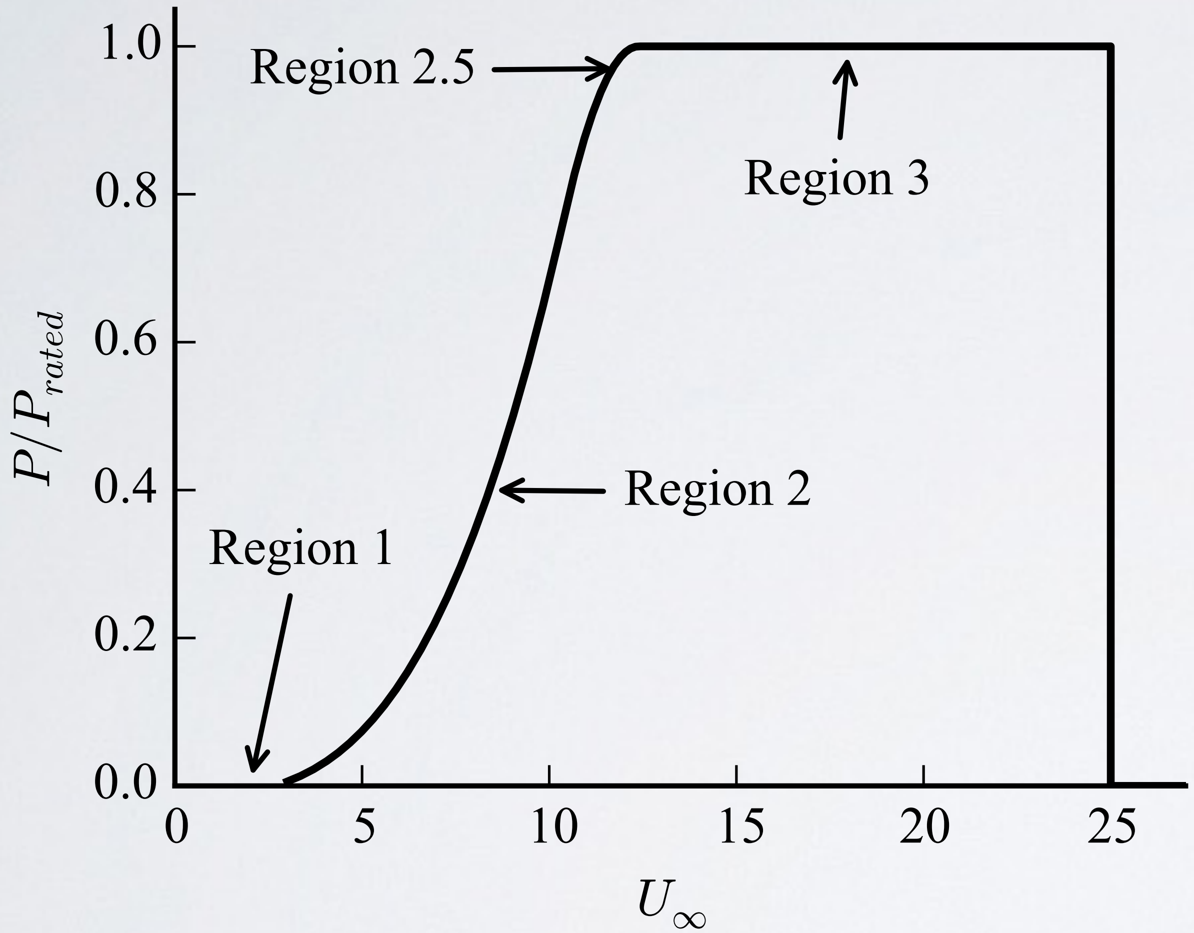


Aerodynamics

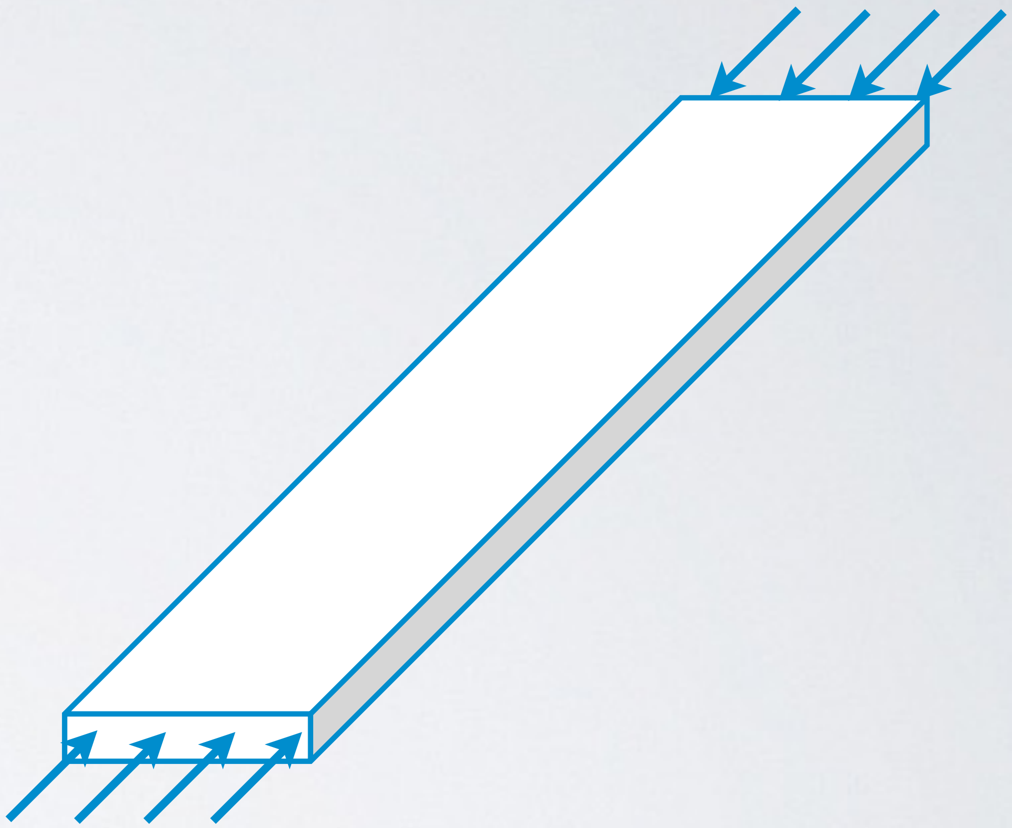


Composite Laminate Theory

Single discipline thinking usually leads to poor solutions

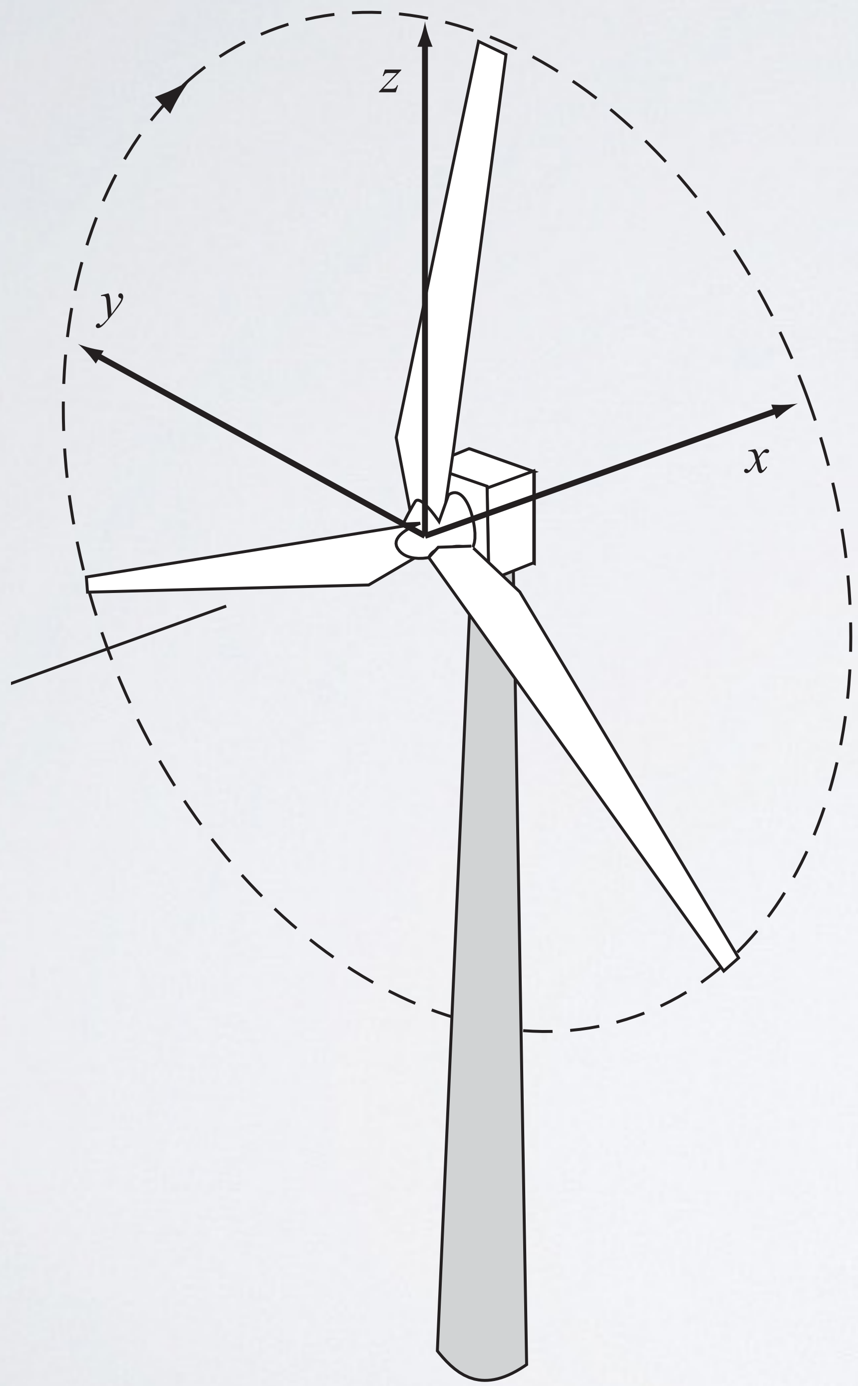


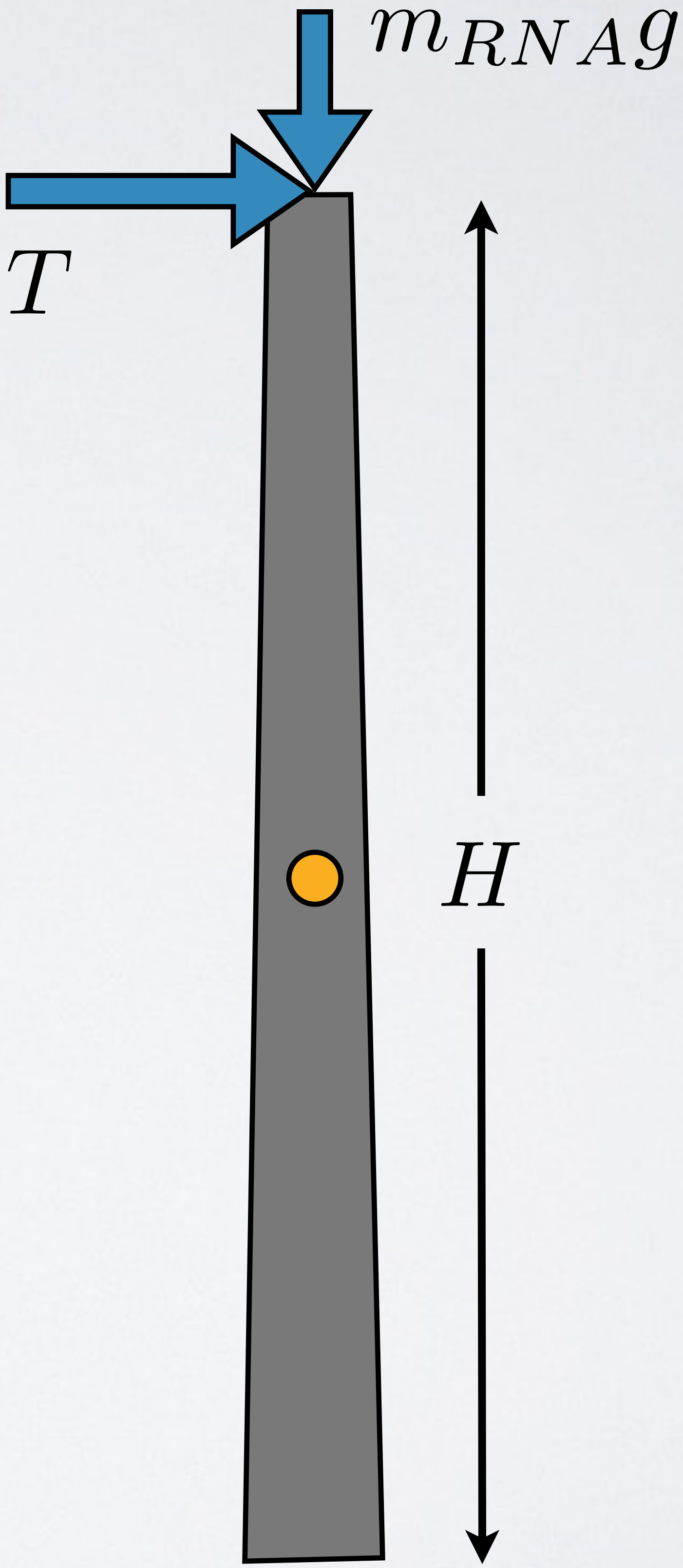
Performance



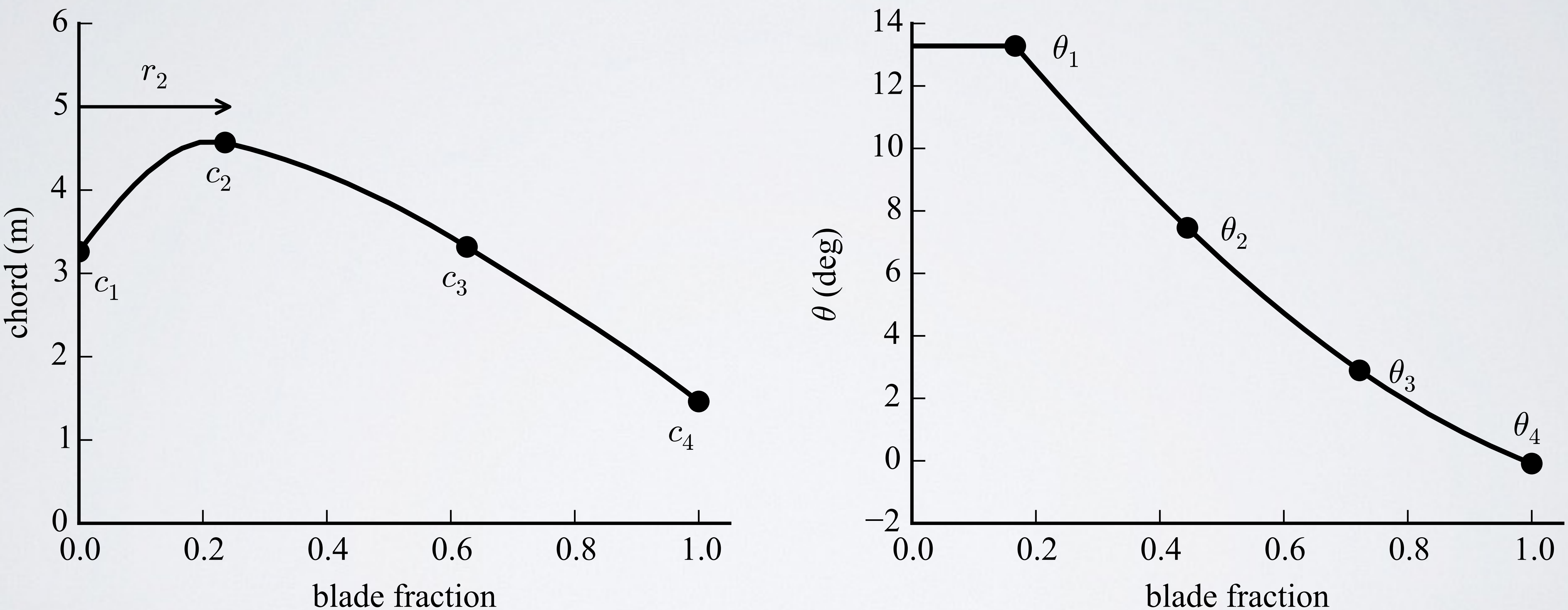
Finite Element Analysis and Buckling

Single *component* thinking usually leads to poor solutions





Splines are an effective way to represent continuous distributions with a small number of design variables



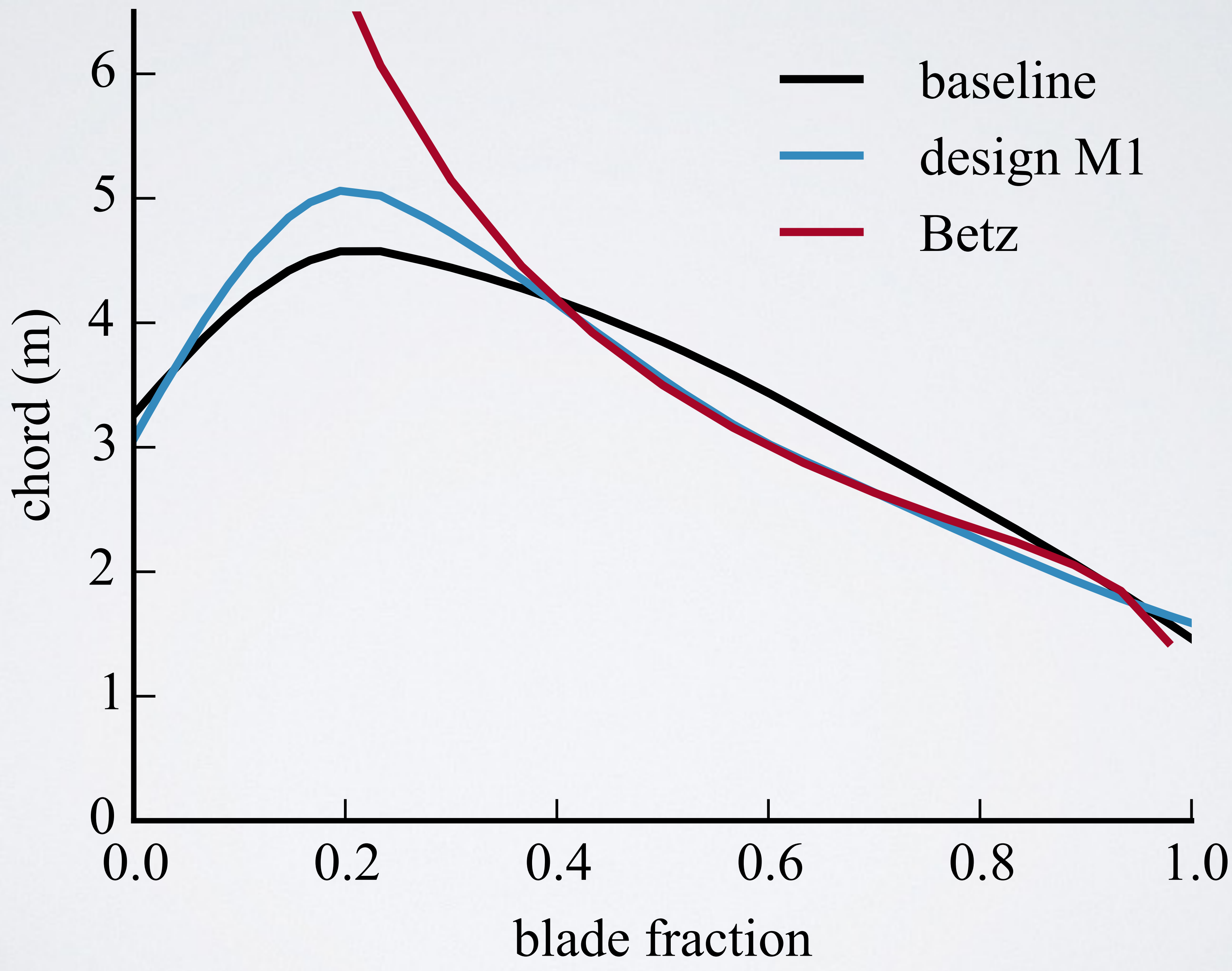
A relatively small number of design variables were used in our early studies

Description	Name	# of Vars
chord distribution	{c}	5
twist distribution	{θ}	4
spar cap thickness distribution	{t}	3
tip speed ratio in region 2	λ	1
rotor diameter	D	1
machine rating	rating	1

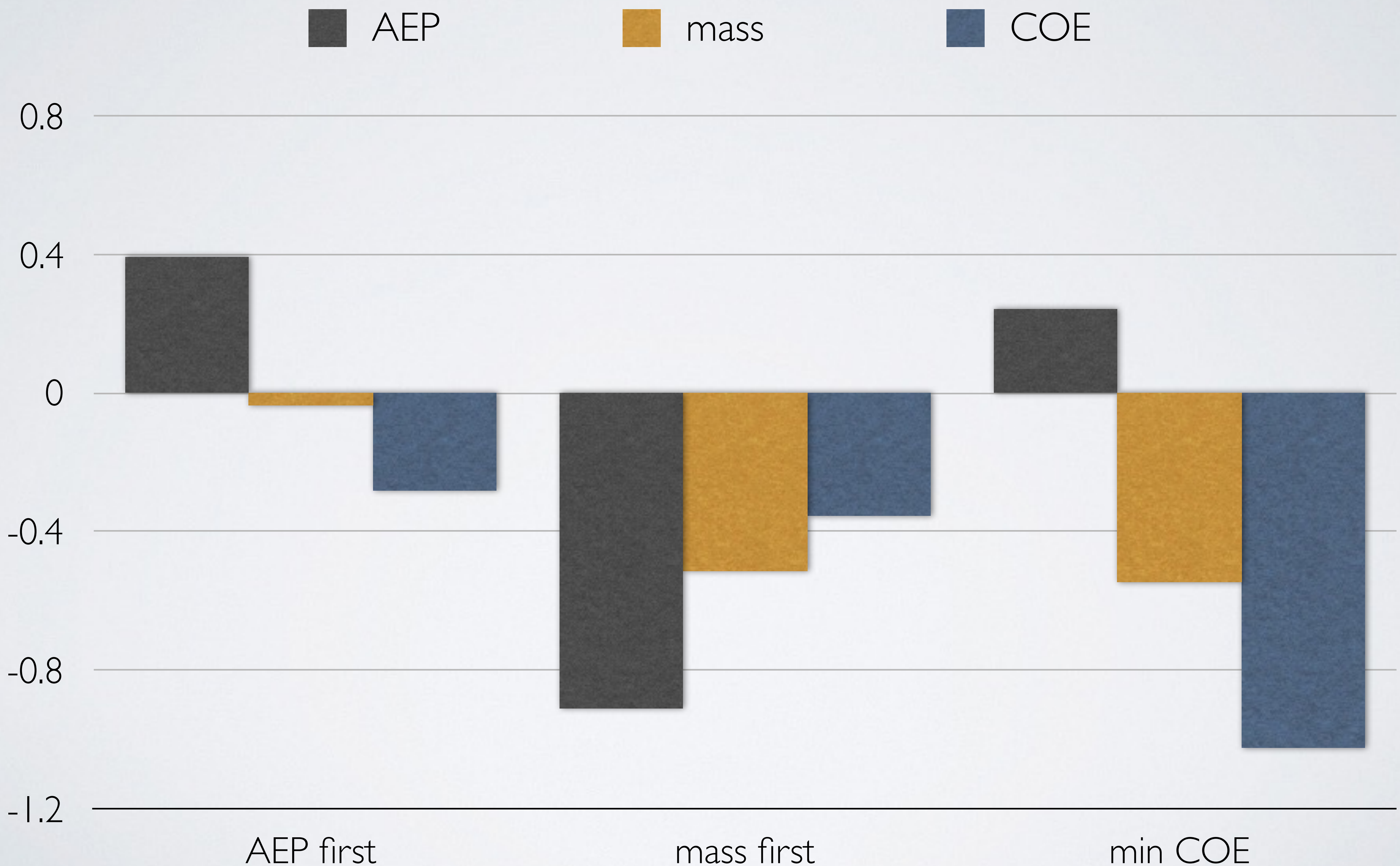
However, we typically used around 100 constraints

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & J(x) \\
\text{subject to} \quad & (\gamma_f \gamma_m \epsilon_{50i})/\epsilon_{ult} < 1, \;\; i = 1, \ldots, N && \text{ultimate tensile strength} \\
& (\gamma_f \gamma_m \epsilon_{50i})/\epsilon_{ult} > -1, \;\; i = 1, \ldots, N && \text{ultimate compressive strength} \\
& (\epsilon_{50j} \gamma_f - \epsilon_{cr})/\epsilon_{ult} > 0, \;\; j = 1, \ldots, M && \text{spar cap buckling} \\
& \delta/\delta_0 < 1.1 && \text{tip deflection at rated} \\
& \omega_1/(3\Omega_{rated}) > 1.1 && \text{blade natural frequency} \\
& \sigma_{\text{root-gravity}}/S_f < 1 && \text{fatigue at blade root (gravity loads)} \\
& \sigma_{\text{root-gravity}}/S_f > -1 && \text{fatigue at blade root (gravity loads)} \\
& V_{tip} < V_{tip_{max}} && \text{maximum tip speed}
\end{aligned}
$$

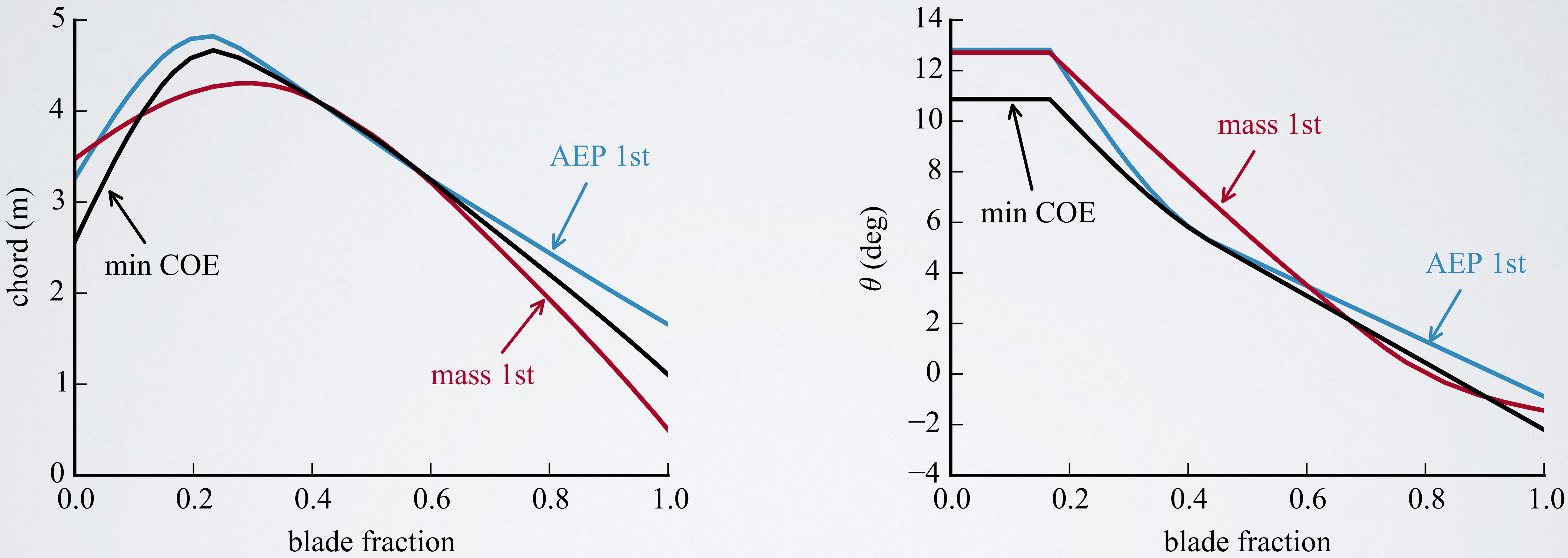
Single discipline optimization lead to inferior results



Even if multiple disciplines were iterated. Integrated optimization is key.



An appropriate objective choice is critical and generally under-appreciated



In higher dimensional space, analytic gradients become increasingly important

Rotor
- Hub Struc
- Rotor Perf
- Rotor Struc
- Rotor Aero
- Blade Struc
- Section Aero
- Section Struc

Nacelle
- LSS/HSS
- Gearbox
- Bearings
- Generator
- Bedplate
- Yaw System

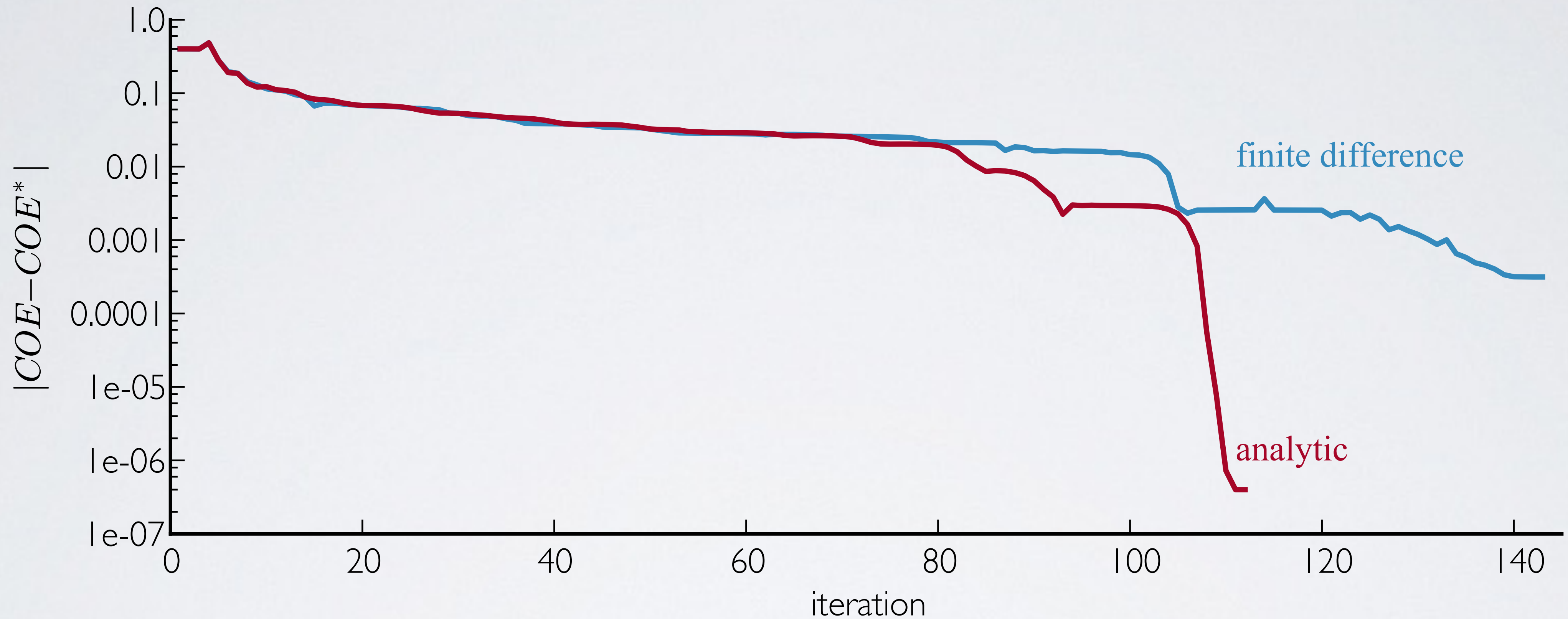
Tower / Foundation
- Tower Struc
- Tower Aero
- Jacket Struc
- Tower Hydro
- Tower Soil

Costs
- TCC
- O&M
- AEP
- BOS
- Finance

In higher dimensional space, analytic gradients become increasingly important

Component	Description	# of vars
Rotor	Chord distribution	5
Rotor	Twist distribution	4
Rotor	Spar-cap thickness distribution	5
Rotor	Trailing-edge panel thickness distribution	5
Rotor	Precurve distribution	3
Rotor	Hub precone angle	1
Rotor	Tip-speed ratio in Region 2	1
Tower	Height	1
Tower	Waist location	1
Tower	Diameter	2
Tower	Shell Thickness	3

As the problem size increases, even finite differencing may not be good enough



	Finite-difference	Analytic
Run time (hours)	5.43	1.11

Some takeaways

- There is a big difference between developing tools just for analysis versus for analysis *and* optimization.
- During analysis development think about gradients, discuss appropriate objectives, and think about the system-level.
- We will discuss later what can/should be done on the optimization side (scaling, multi-start, reformulation, etc.)

Some optimizers you might be interested in

- fmincon: Matlab, 4 algorithms
- SNOPT: commercial tool from Stanford. talk to me about license
- OptdesX, APOPT: tools available at BYU
- scipy.optimize: Python, not great, but easy to use
- KNITRO: academic version
- CVX, Gurobi: convex optimization, Matlab-based
- CPLX: linear and integer programming

Some *frameworks* you might be interested in

- pyOptSparse: Python, a wrapper to many optimizers
- OpenMDAO: Python-based, developed at NASA, not a "black box" approach, coupled derivatives, MDO architectures, HPC support
- ModelCenter or Isight: tool to integrate models with interfaces to other tools like Matlab, Excel, etc.
- DAKOTA: Sandia, only has open-source optimizers, but allows easy coupling to UQ algorithms
- AMPL: a mathematical programming language, supports many optimizers