MATH-329
Software demo

Course feedback is open – your input matters.

I'll post info about the **final exam** format shortly (incl. list of proofs).

With the right tools and some experience,
the programming parts in HW1, HW2, HW3 take less than an hour.

Now that you learned how to do everything by hand,
you can be a well-informed user of these tools,
because you have a sense of how they work.

We use a few different optimization tools

Matlab's Optimization Toolbox (built-in if you selected it when installing Matlab)

`fminunc`, `fmincon`, `linprog`, `quadprog` <- type '`doc xyz`' for help

CVX: a modeling language for convex optimization

http://cvxr.com/cvx/download/ <- website has documentation

> `cvx_setup`

Manopt: a toolbox for optimization on smooth sets (including linear spaces $\mathcal{E}$)

https://www.manopt.org/downloads.html <- website has documentation

https://github.com/NicolasBoumal/manopt

> `importmanopt`

In Python: CVXPY, PyManopt, JAXopt, SciPy, …

Algorithms are designed for problem classes

Hence users need to

1. identify the right tool for their problem, and
2. describe their problem in a format the algorithm understands.

- `fminunc`: $\min_{x \in \mathbf{R}^n} f(x)$ -- works also with variable in $\mathbf{R}^{m \times n}$ or other arrays.
- `fmincon`: $\min_{x \in \mathbf{R}^n} f(x)$ s.t. $Ax \leq b, \tilde{A}x = \tilde{b}, h(x) = 0, g(x) \leq 0, \ell \leq x \leq u$
- `linprog`: $\min_{x \in \mathbf{R}^n} c^\top x$ s.t. $Ax \leq b, \tilde{A}x = \tilde{b}, \ell \leq x \leq u$
- `quadprog`: $\min_{x \in \mathbf{R}^n} \frac{1}{2} x^\top H x + c^\top x$ s.t. $Ax \leq b, \tilde{A}x = \tilde{b}, \ell \leq x \leq u$

User should also provide derivatives (e.g., ∇f, $\mathrm{D}h$, $\mathrm{D}g$) if useful for algorithms.

Algorithms are designed for problem classes

Hence users need to

1. identify the right tool for their problem, and
2. describe their problem in a format the algorithm understands.

- CVX: $\min_{x \in S} f(x)$ where S and f are convex,
  both described in such a way that the toolbox *knows* they are convex and can exploit it. CVX is a modeling language, not an algorithm.

- Manopt: $\min_{x \in \mathcal{M}} f(x)$ where $\mathcal{M}$ is a Riemannian manifold (MATH-512).
  In particular, $\mathcal{M}$ can be any Euclidean space $\mathcal{E}$: convenient to go beyond $\mathbf{R}^n$, $\mathbf{C}^n$.

These tools can use automatic differentiation, at some cost for efficiency.

`fminunc` example

$$\min_{X \in \mathbf{R}^{n \times n}} \|X^2 - A\|^2$$

```
n = 2;
A = randn(n, n); A = (A+A')/2 + 2*n*eye(n);
sqnorm = @(M) M(:)'*M(:);  % sqnorm(M) = ||M||^2
X0 = randn(n, n);
X = fminunc(@(X) sqnorm(X*X-A), X0);
```

To display info: `options = optimoptions('fminunc', 'Display', 'iter');`

Can also supply the gradient / Hessian of cost function with extra code.

Manopt example

$$\min_{X \in \mathbf{R}^{n \times n}} \|X^2 - A\|^2$$

```
n = 2;
A = randn(n, n); A = (A+A')/2 + 2*n*eye(n);
sqnorm = @(M) M(:)'*M(:);  % sqnorm(M) = ||M||^2
problem.M = euclideanfactory(n, n);
problem.cost = @(X) sqnorm(X*X - A);
problem = manoptAD(problem);
X = trustregions(problem);
```

Instead of AD, could also set `problem.grad` and `problem.hess`.

CVX example

$$\begin{array}{ll} \text{minimize} & \|Ax - b\|_2 \\ \text{subject to} & Cx = d \\ & \|x\|_\infty \leq e \end{array}$$

```
m = 20; n = 10; p = 4;
A = randn(m,n); b = randn(m,1);
C = randn(p,n); d = randn(p,1); e = rand(1);
cvx_begin
    variable x(n)
    minimize( norm( A * x - b, 2 ) )
    subject to
        C * x == d
        norm( x, Inf ) <= e
cvx_end
```

Let’s look at some simple problems

For each one, think:

What kind of problem is this?

constrained / unconstrained; convex / nonconvex; … ?

Which tools could I use?

Matlab’s built-ins, CVX, Manopt, others?

Exercise week 1: smallest enclosing ball

Find the smallest ball that contains k given points $x_1, \ldots, x_k \in \mathbf{R}^n$.

Can be formulated as: Find center x and minimal radius r solving

$$\min_{x \in \mathbf{R}^n, r \in \mathbf{R}} r \qquad \text{subject to} \qquad \|x - x_i\| \leq r \text{ for } i = 1, \ldots, k$$

This is a convex problem: well suited for CVX. Could also use fmincon.

Can even use fminunc after reformulation; how?

Exercise week 1: linear program

n power plants need to provide electricity to m cities.

Power plant i produces at most w_i MWh. City j needs d_j MWh.

Sending energy from plant i to city j costs C_{ij} CHF/MWh.

Minimize the cost of energy distribution to meet the demand.

Variable: $X \in \mathbf{R}^{n \times m}$ with X_{ij} the number of MWh plant i sends city j.

$$\min_{X \in \mathbf{R}^{n \times m}} \sum_{\substack{i=1\dots n \\ j=1\dots m}} C_{ij} X_{ij} \qquad \text{subject to} \qquad X \geq 0$$

$$\sum_{i=1\dots n} X_{ij} \geq d_j \text{ for all } j$$

$$\sum_{j=1\dots m} X_{ij} \leq w_i \text{ for all } i$$

Linear program: approach 1

Use Matlab's builtin `linprog`: a solver for linear programming.

$$\min_{X \in \mathbf{R}^{n \times m}} \sum_{\substack{i=1\ldots n \\ j=1\ldots m}} C_{ij} X_{ij} \quad \text{subject to} \quad X \geq 0$$

$$\sum_{i=1\ldots n} X_{ij} \geq d_j \text{ for all } j$$

$$\sum_{j=1\ldots m} X_{ij} \leq w_i \text{ for all } i$$

How? Type '`help linprog`'. Great for CPU time. Bad for *your* time.

Linear program: approach 2

Use CVX: a modeling language translates your description for a solver.

$$\min_{X \in \mathbf{R}^{n \times m}} \sum_{\substack{i=1\ldots n \\ j=1\ldots m}} C_{ij} X_{ij} \quad \text{subject to} \quad X \geq 0$$

$$\sum_{i=1\ldots n} X_{ij} \geq d_j \text{ for all } j$$

$$\sum_{j=1\ldots m} X_{ij} \leq w_i \text{ for all } i$$

See code. Takes just fifteen minutes to write with some experience.

Exercise week 2: Rosenbrock function

With parameters $a, b > 0$, define $f: \mathbf{R}^2 \to \mathbf{R}$ as:

$$f(x) = (a - x_1)^2 + b(x_2 - x_1^2)^2$$

Set $a = 1, b = 100$. Setup algorithm to find minimizer (it's at $(1,1)$).

Approach 1: Matlab's builtin `fminunc`.

Approach 2: Manopt.

Exercise week 7: optimizing on a circle or disk

$$\min_{x \in \mathbf{R}^2} x_1 + x_2 \quad \text{subject to} \quad x_1^2 + x_2^2 = 1$$

Approach 1: Matlab's `fmincon`.

Approach 2: Manopt (because the circle is a "nice, smooth set").

$$\min_{x \in \mathbf{R}^2} x_1 + x_2 \quad \text{subject to} \quad x_1^2 + x_2^2 \leq 1$$

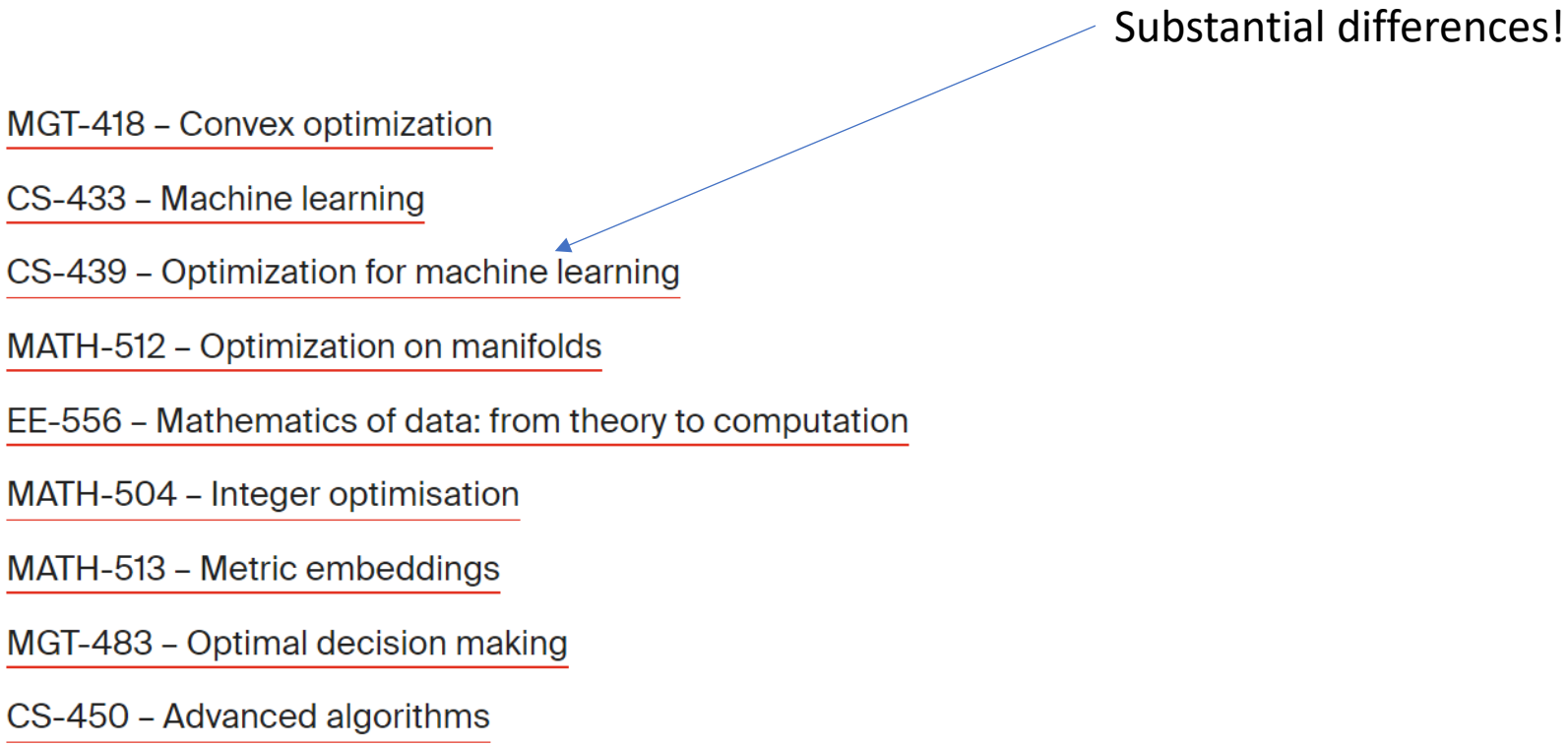
Approach 1: Matlab's `fmincon`.

Approach 2: CVX (because this happens to be a convex problem).

Optimization at EPFL

EPFL offers many courses about / using optimization.

Masters level:



https://www.epfl.ch/research/domains/optimization/home/courses/