Decomposition methods

imagine we have to solve

$$\min \quad f_1(x_1, y) + f_2(x_2, y)$$

f_1, f_2 convex, $\underline{x}_1, \underline{x}_2$ big

not essential

We could attack this by

$$\min_y \left[\left[\min_{x_1} f_1(x_1, y)\right] + \left[\min_{x_2} f_2(x_2, y)\right]\right]$$

Which has the advantage that the inner problems can be done in ~~are~~ parallel.

Process:

fix some y_0

→ solve $\phi_1 = \min_{x_1} f_1(x_1, y_n)$

$\phi_2 = \min_{x_2} f_2(x_2, y_n)$

generate y_{n+1}

Usually thought of as a master problem and two (some) slave problems

slaves: $\phi_1(y) = \min_{x_1} f_1(x_1, y)$

$\phi_2(y) = \min_{x_2} f_2(x_2, y)$

master: $\min_y \phi_1(y) + \phi_2(y) = \phi(y)$.

Now depending on f_1, f_2, master could have a variety of forms. Assume f_1, f_2 convex, smooth – then ϕ_i smooth, convex

So we could do gradient descent on ϕ.

Procedure:

choose y_0

– Solve slaves, return

$$\left.\frac{\partial f_i}{\partial y}\right|_{x_i, y_n}$$

$$- \; y_{n+1} = y_n - h \nabla \phi = y_n - h\left[\left.\frac{\partial f_1}{\partial y}\right|_{x_1, y_n} + \frac{\partial f_2}{\partial y}\right.$$

Clearly works for more than one slave.

Notice because this is a form of coordinate ascent, there can be trouble

$$f_1(x_1, y) = (x+y)^2 + \varepsilon(x-y)^2$$
$$f_2(x_2, y) = (x-y)^2 + \varepsilon(x+y)^2$$

$y_0 = 1$

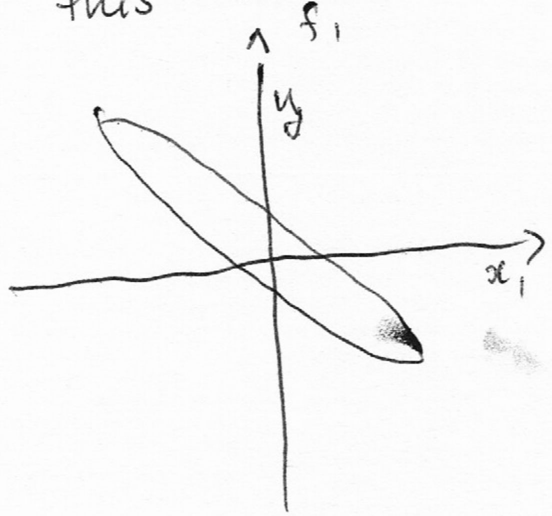
then $\quad x_1 = -\left(\frac{1-\varepsilon}{1+\varepsilon}\right)$

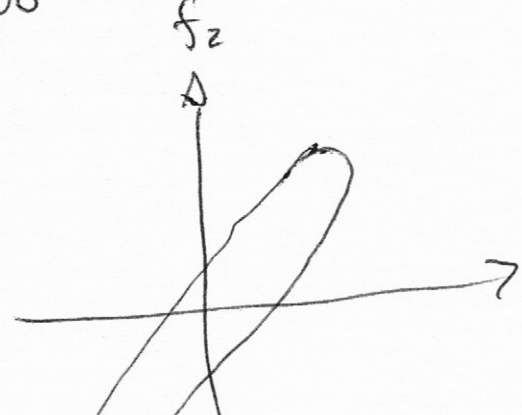
$\qquad\quad x_2 = \left(\frac{1-\varepsilon}{1+\varepsilon}\right)$

So $\phi(y) = \cancel{(1+\varepsilon)}\, 2\left(y - \frac{1-\varepsilon}{1+\varepsilon}\right)^2 + 2\varepsilon\left(y + \left(\frac{1-\varepsilon}{1+\varepsilon}\right)\right)^2$

and min is at $y \sim 1 - 4\varepsilon + O(\varepsilon^2)$.

$\rightarrow$ this means slow steps





<u>Alternative</u>:

$$\min \ f_1(x_1, y_1) + f_2(x_2, y_2)$$

$$\text{st} \quad y_1 = y_2$$

Consider Dual

Lagrangian is $\mathcal{L}(\lambda, x_1, y_1, x_2, y_2)$

$$= f_1 + f_2 + \lambda(y_1 - y_2)$$

<u>Dual is</u>

$$g(\lambda) = \inf_{x_1, y_1, x_2, y_2} \mathcal{L}(\lambda, x_1, y_1, x_2, y_2)$$

and dual is a <u>lower</u> bound on problem

$$V(x_1, y_1, x_2, y_2) \geq g(\lambda)$$

So now we can do subgradient <u>ascent</u> on dual to max dual

<u>Note</u> dual is envelope of linear functions it ~~may be~~ is convex, but may not be smooth

So

- λ^0, x_1^0, y_1^0 etc

Slaves:

$$\min_{x_1, y_1} f_1(x_1^*, y_1^*) + \lambda^n y_1$$

$\vdots$

Master:

$$\partial g(\lambda) = y_1^{n+1} - y_2^{n+1}$$

$$\lambda^{n+1} = \lambda^n + h_n \partial g(\lambda)$$

Notice
- when we have evaluated slaves, we get a value for the primal
- master gives us a value for dual
- if we happen to encounter $x_1, y_1, x_2, y_2, \lambda$ st $V(x_1, y_1, x_2, y_2) = g(\lambda)$

this point is a soln:

Notice we can deal w/ probs coupled by <u>constraints</u>

$$\min \; f_1(x_1) + f_2(x_2)$$

$$\text{st} \quad h_1(x_1) + h_2(x_2) \leq 0$$

write Lagrangian

$$L(x_1, x_2, \lambda) = f_1(x_1) + \lambda^T h_1(x_1) + f_2(x_2) + \lambda^T h_2(x_2)$$

<u>Dual is</u>:

$$g(\lambda) = \inf_{x_1, x_2} L(x_1, x_2, \lambda).$$

so we max dual

notice for λ^* some value of λ, then

$\cancel{\partial g \big|_{\lambda^*}}$ write x_1^*, x_2^* for

$\cancel{\text{that}}$ $\operatorname{argmin}_{x_1} f_1(x_1) + \lambda^* h_1(x_1)$

$\operatorname{argmin}_{x_2} f_2(x_2) + \lambda^* h_2(x_2)$

then $\partial g \big|_{\lambda^*} = h_1(x_1^*) + h_2(x_2^*)$

So our procedure becomes

- Start w/ some λ_0
- Slaves

$$\underset{x_i}{\operatorname{argmin}}\ f_i(x_i) + \lambda_n h_i(x_i)$$

- Master

$$\lambda_{n+1} \rightarrow \lambda_n + h_n \left[h_1(x_1^*) + h_2(x_2^*) \right]$$

if we encounter x_1^*, x_2^*, λ^* st

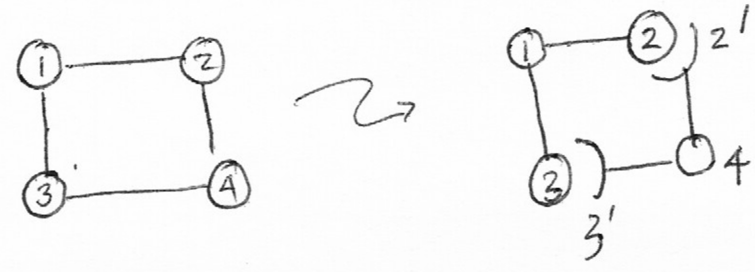
$$f_1(x_1^*) + f_2(x_2^*) = g(\lambda^*)$$

we are done.

We can apply this strategy to discrete problems, too (main issue is notation)

• <u>Simple example</u>

<u>MRF</u>:



<u>equiv</u>

$$\sum_i \theta_i(x_i) + \theta_{12}(x_1,x_2) + \theta_{13}(x_1,x_3) + \theta_{24}(x_2,x_4) + \theta_{34}(x_3,x_4)$$

$$x_i \in \cancel{\{\text{labels}\}}\ \{0,1\}$$

$$\sum_i \theta_i(x_i) + \theta_{12}(x_1,x_2) + \theta_{13}(x_1,x_3) + \theta_{34}(x_3',x_4) + \theta_{24}(x_2',x_4)$$

S.T. $x_3 = x_3'$, $x_2 = x_2'$

$x_i \in$ ~~labels~~. $\{0,1\}$

Now for decomposed problem

$$\mathcal{L} = C_{T_1} + C_{T_2} + \lambda_3(x_3 - x_3') + \lambda_4(x_4 - x_4')$$

<u>Notice</u>: Each <u>component</u> is a tree

- So <u>slaves</u> are easy
- Notice lms change the unary terms

For more interesting cases, we need to construct some notation

- For consistency w/ Sontag et. al. review, I will max
- Divide problem into unary terms and $(k>1)$-ary terms, called <u>factors</u>
- each factor gets its own copy of each variable that appears in the factor
- <u>vars</u>: $x_i \leftarrow$ i'th variable, version for unary term

  $x_i^f \leftarrow$ i'th variable, copy that lives in factor f

  $x_f \leftarrow$ all the vars for factor f, (that factor's copies)

With that notation, we can write

$$\max_{x_i,\, x_f^{i}} \quad \sum_i \theta_i(x_i) + \sum_f \theta_f(x_f)$$

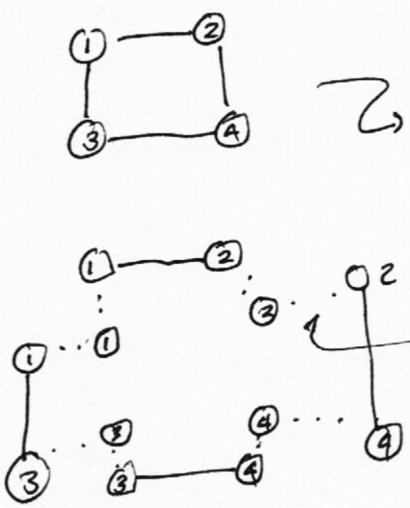
$$\text{s.t.} \quad \mathbb{1}(x_i = l) - \mathbb{1}(x_i^f = l) = 0 \qquad \forall i, f, l$$

indicator function

label

(Notice how this deals w/ non 0-1 labels)

eg



constraints

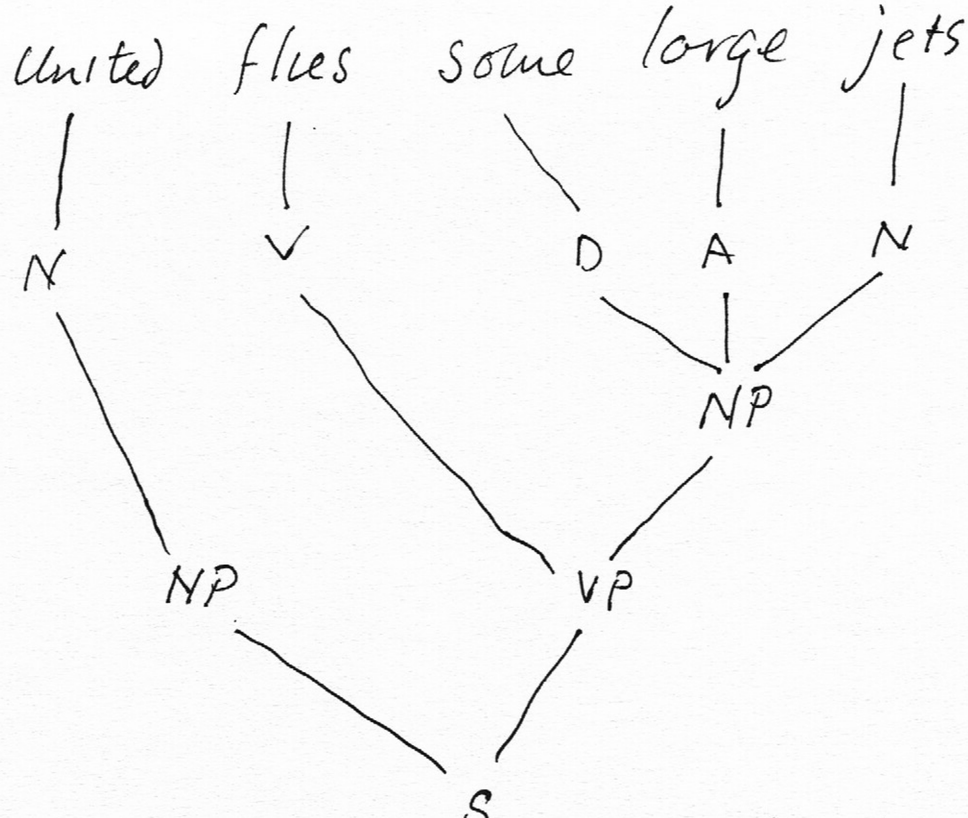
This means we have rather a lot of lagrange multipliers
(one per pair of states for each constraint link)

- This justies seeing the L.M.'s as messages
- You can see a link to D.P. fairly clearly here.
- Issues
  - when to stop?
    - if you encounter a soln (Primal = Dual)
    - No progess on Dual
      - which x_i values?
        - at random
        - in some cases, fair approx is obvious (follows
  - How to decompose?
    - per problem.

Example: making parse trees and P.O.S tags consistent

(Rush + Collins)

· Parsing



· there are methods to construct such trees by

$$y^* = \underset{y \in \text{Trees}}{\operatorname{argmax}} \, f(y)$$

Tagging:

United flies some large jets

N — V — D — A — N
oun erb eterminer djective oun.

- there are methods to produce a set of tags by

$$z^* = \arg\max_{z \in Tags} g(z)$$

- Now imagine we want to make the tags consistent with the parse.

- Define $l(y) = \{$strip tree from y, produce tag sequence$\}$

- then we are interested in

$$\arg\max_{y \in Trees,\, z \in Tags} f(y) + g(z)$$

$$\text{s.t.} \quad z = l(y)$$

We know how to deal with this

$$L(\delta, y, z) = f(y) + g(z) + \delta^T [z - \ell(y)]$$

$$\Gamma(\delta) = \sup_{y, z} L(\delta, y, z).$$

and $\Gamma(\delta) \geq$ value of primal

so $\min \Gamma(\delta)$

$\rightarrow$ this gives us slaves

$$\operatorname{argmax}_y f(y) - \delta^T \ell(y)$$

$$\operatorname{argmax}_z g(z) + \delta^T z$$

$\rightarrow$ master $\leftarrow$ subgradient is easy.

$$\min \Gamma(\delta)$$

<u>IDEA</u>: - force agreement between multiple parsing models