1. General LP duality: basics of optimal transport (Becker model)

* LP solvers (o.w. Gurobi, simpler black-box than hand-coded simplex from IBM for instance)

1. Incorporate unobserved heterogeneity (logit)

* Entropic optimal transport (with Sinkhorn’s algorithm)

1. Inverse optimal transport: estimation of matching models

* Generalized linear models (with scikit-learn): first directly from utilities and productivities, then conversely to recover structural parameters (complicated due to lack of control group in AKM??)

1)

a) Diet problem: j=food, i=nutrient, N\_ij = quantity of nutrient I brought by one unit (1$) of food j, r\_i = daily requirement of nutrient I, q\_j = $ amount spent on food j

* min\_q SUM(q\_j) s.t. SUM(N\_ij\*q\_j) >= r\_i
* min\_q q^T\*c with c\_j=1 s.t. N\*q >= r
* V\_p = min\_q [q^T\*c + max\_pi SUM\_i(pi \* (r\_i – SUM\_j(N\_ij\*q\_j)))] is the primal problem : the trick is that we want to penalize up to infinity when the constraint is not met!
* V\_p = min\_q max\_pi [q^T\*c + pi^T\*r – pi^T\*Nq] >= max\_pi [pi^T\*r + min\_q q^T\*(c-N^T\*pi)] = V\_D : strong duality / minmax theorem (we recover a penalty term pushing to minus infinity when constraint does not hold) holds if a feasible solution exists (might not with equality constraints or negative N matrix), else just the weak version.
* V\_D = max\_pi pi^T\*r s.t. N^T\*pi <= c with pi\_i interpreted as shadow price of nutrient i (some food cannot be cheaper than the sum of its nutrients)

NB: Get free license for Gurobi when a student! Interior-point method + linear constraints + quadratic programming solvers performs better than mere simplex or scipy…

NB: By convention, the primal is often presented as a max problem. In reality, it refers to central planner as opposed to decentralized solution (dual). Compared to nonlinear programming, here we have corner solutions!

NB: We can select solutions from the primal (for instance) by solving a secondary LP problem or by adding a regularization term in the primary problem (see later for entropy objective)!

b) New problem: 2 populations of workers with characteristics x in X, o.w. there are n\_x of type x (later interpreted as frequency in continuous model) + firms of type y in Y, with numbers m\_y.

In general, SUM\_x(n\_x) </> SUM\_y(m\_y). Here, we do not consider outside options and do not enforce matching for everybody. If x and y match, worker’s utility (in monetary terms) is alpha\_xy and firm’s production is gamma\_xy. The value of an xy match is PHI\_xy = alpha\_xy + gamma\_xy (exogenous). We look for the optimal matching: a matching mu\_xy is the number of xy matches formed (under obvious constraints), it is optimal from the central planner’s perspective when max\_mu SUM\_xy(mu\_xy\*PHI\_xy) s.t. SUM\_y(mu\_xy)<=n\_x & SUM\_x(mu\_xy)<=m\_y. If there is no match, utility/production is zero.

Again, this is a primal LP problem V\_P. We write the Lagrangian:

V\_P = max\_mu [SUM\_xy(mu\_xy\*PHI\_xy) + min\_u\_x SUM\_x(u\_x\*(n\_x – SUM\_y(mu\_xy))) + min\_v\_y SUM\_y(v\_y\*(m\_y – SUM\_x(mu\_xy)))]

We get the dual:

V\_D = min\_u+v SUM\_x(n\_x\*u\_x) + SUM\_y(m\_y\*v\_y) + max\_mu\_xy SUM\_xy(mu\_xy\*(PHI\_xy – u\_x – v\_y)) (last term equivalent to constraint u\_x + v\_y >= PHI\_xy).

Now, we have 2 interpretations of the dual: stable matching (no blocking pairs) and competitive wage equilibrium. Indeed, u\_x and v\_y can be interpreted as the respective gains of workers and firms, which must be greater than what they would get by matching among themselves (else there always exists a way to split the associated surplus to satisfy both parts). Also, the fact that we minimize individual utilities has to do with the competitive market solution, excluding the possibility of rents (v\_y = max\_x(PHI\_xy – u\_x))… This gives the wage interpretation (Becker model). Note that this work with transferable utility (cf. also marriage market).

Rewriting the problem from a wage determination (on top of utility) perspective:

U\_x = max\_y [alpha\_xy + w\_xy, 0] & v\_y = max\_x [gamma\_xy – w\_xy, 0] yield general equilibrium wage solution s.t. gamma\_xy – v\_y <= w\_xy <= u\_x – alpha\_xy.

Also do not forget complementary slackness! If mu is an optimal primal solution, and (u,v) an optimal dual solution, then mu\_xy > 0 🡪 u\_x + v\_y = PHI\_xy, u\_x>0 🡪 SUM\_y(mu\_xy)=n\_x, v\_y>0 🡪 SUM\_x(mu\_xy) = v\_y. This can be interpreted as a splitting rule for wage determination and no unmatched condition.

c) Difficulty compared to standard LP is that we need to vectorize matrices with Kronecker products. From the primal, we can write the argument matrix with row-major order (stack rows, default in numpy) or column-major order (stack columns). We can rewrite the primal as:

max\_mu vec\_R(mu)^T \* vec\_R(PHI) s.t. mu\*1\_y<=n & 1\_x^T\*mu<=m

Kronecker product yields : A (\*) B = (a\_11\*B, a\_12\*B ; a\_21\*B, a\_22\*B) (definition is not symmetric, but could use tensor product?)

Properties are: vec\_R(A\*X\*B^T) = A (\*) B \* vec\_R(X)

Therefore, constraints turn to: I\_x (\*) 1\_y^T \* vec\_R(mu) <=n & 1\_x^T (\*) I\_y \* vec\_R(mu) <=m

We write M = (I\_x (\*) 1\_y^T ; 1\_x^T (\*) I\_y) and give all that to LP solver (Gurobi)! This is a sparse (margining-out) matrix, hence more efficient algorithms…

2)

a) See crash course on logit model: we can add a scaling factor to modulate the amount of heterogeneity on the Gumbel error term. Also remember that the ex-ante indirect utility is just the log of the logit term denominator. **Microfoundation?**

Back to worker’s problem: we want to determine endogenous quantities w\_xy and mu\_xy. We have:

U\_x = E[max\_y [alpha\_xy + w\_xy + sigma\*epsilon\_y, 0]]

V\_y = E[max\_x [gamma\_xy – w\_xy + sigma\*nu\_x, 0]]

.. so as to rationalize different choices made by similar workers: Choo-Siow = Becker + McFadden

* Theoretically, Gumbel beats other distributions (limit distribution of a maximum), but also empirically, as other distributions can be included in the framework (See *Galichon & Salanié, Cupid’s invisible hand, Restud 2022*)! Sigma can also be heterogeneous, etc…

Then: P(y|x) = exp((alpha\_xy + w\_xy / sigma)) / exp(U\_x/sigma) (from log-formulation of expected utility) = exp((alpha\_xy + w\_xy – u\_x)/sigma) = mu\_xy / n\_x in equilibrium!

We also (for unemployment, important to have a baseline for identification because of fixed effect floating) have mu\_x0/n\_x = P(0|x) = exp(-U\_x/sigma).

Hence: mu\_xy/mu\_x0 = exp((alpha\_xy + w\_xy)/sigma).

Reciprocally, on the firm’s side: mu\_xy/mu\_0y = exp((gamma\_xy – w\_xy)/sigma).

We get rid of w\_xy by taking: mu\_xy² = mu\_x0\*mu\_0y\*exp(PHI\_xy/sigma) (Choo-Siow for separability).

Then, by substituting into the constraints:

mu\_x0 + SUM\_y(sqrt(mu\_x0\*mu\_0y)\*exp(PHI\_xy/(2\*sigma))) = n\_x

mu\_0y + SUM\_x(sqrt(mu\_x0\*mu\_0y)\*exp(PHI\_xy/(2\*sigma))) = m\_y

* Transferable utility with additive forms implies a linear relation between u\_x and v\_y. All other functional forms relate to imperfect transferable utility (see Galichon et al., JPE 2019)

We can solve by initially setting mu\_0y^0 = m\_y (or conversely?), then iterate over to solve for mu\_x0 and mu\_y0 (mu\_xy is a function of those).

* In optimal transport, there is no one unmatched, but the algorithm works under its Sinkhorn version.

b) It is going to converge to an existent and unique solution, and we can show it with duality:

Max SUM\_xy(mu\_xy\*PHI\_xy) – sigma\*epsilon(mu) = min SUM\_x(n\_x\*u\_x) + SUM\_y(m\_y\*v\_y) + 2\*sigma\*SUM\_xy(sqrt(n\_x\*m\_y)\*exp((PHI\_xy – u\_x – v\_y)/(2\*sigma))) + sigma\*SUM(exp(-u\_x/sigma)) + sigma\*SUM\_y(exp(-v\_/sigma))

..with epsilon(mu) = … (from hard to soft (entropic) penalization, see Cupid’s paper for reference)

Now, the dual is unconstrained (we could also do the primal). We get FOC:

N\_x = SUM\_y(exp((PHI\_xy – u\_x – v\_y)/(2\*sigma))) + exp(-u\_x/sigma) = SUM\_y(mu\_xy) + mu\_x0

M\_y = SUM\_x(mu\_xy) + mu\_y0

We see that the algorithm corresponds to a coordinate descent algorithm on a convex problem !!

c) Computation pb is that exponentials explode when sigma is small: we use the log-sum-exp trick (see notebook). Numerically, since we get weakly negative terms in the exponentials, we fix the issue.

When sigma is large, we just apply the above version…

We can also think of the FOC as a GLM

3)

PHI\_xy^lambda = SUM\_k(lambda\_k \* PHI\_xy^k): let us estimate lambda.

We rewrite the dual as: W(lambda) = …

We look for lambda such as the predicted matching patterns fit the observed patterns, more precisely their moments. Basically, PHI subsume observable characteristics, and lambda allows to account for unobservables.

Differential of the dual yields (envelope theorem):

SUM\_xy(sqrt(n\_x\*m\_y)\*exp((PHI\_xy^lambda-u\_x-v\_y)/(2\*sigma))\*PHI\_xy^k)

This actually yields a FOC when considering the difference between W and its target in the programme. Taking MSE would not be convex. In the end, it is just a weighted Poisson regression to be estimated with gradient descent (?), not even a method of moments! See notebook for math.

PI(z|theta) = (theta^z)\*exp(-theta)/z\_theta’ 🡪 l(theta) = SUM-i(z\_i\*log(theta)) - theta - …

This yields a Poisson regression corresponding to the objective: …

NB: The observed covariates are PHI^k and fixed effects! Need to identify parameters with Poisson formulation

NB: We could use JAX with the smooth gradient descent (no need for Gurobi)!

Going closer to AKM with identification from movers? Also, we are explicit about the match values, but do not disentangle between individual and match specific components…