

Convex Network Flows

Theo Diamandis

MIT (Julia Lab) & Bain Capital

joint work with Guillermo Angeris and Alan Edelman

JuMP-dev 2024

Goals

- ▶ Goal: solve decision-making problems involving very large nonlinear networks

Goals

- ▶ Goal: solve decision-making problems involving very large nonlinear networks
 - Natural way to specify the relevant optimization problem

Goals

- ▶ Goal: solve decision-making problems involving very large nonlinear networks
 - Natural way to specify the relevant optimization problem
 - Fast and parallelizable computational building blocks

Goals

- ▶ Goal: solve decision-making problems involving very large nonlinear networks
 - Natural way to specify the relevant optimization problem
 - Fast and parallelizable computational building blocks
- ▶ This talk: marriage of *convex optimization* and *network flows*

Outline

Motivation

Framework

Applications

Algorithm

Outline

Motivation

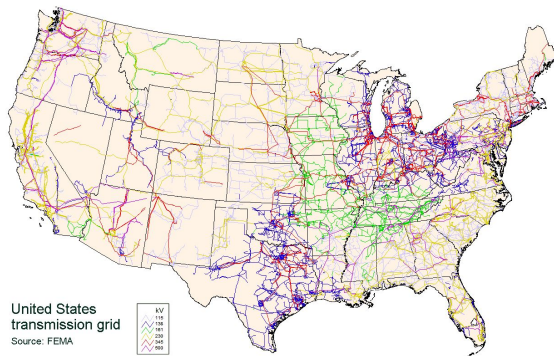
Framework

Applications

Algorithm

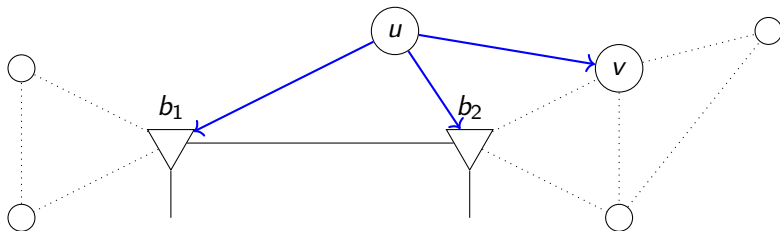
Nonlinear networks are everywhere!

► Power dispatch



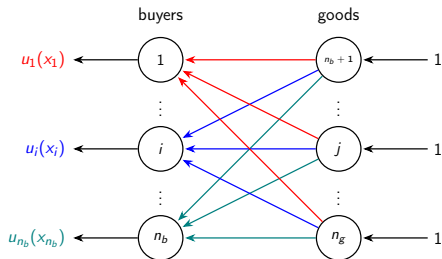
Nonlinear networks are everywhere!

- ▶ Power dispatch
- ▶ Communication systems



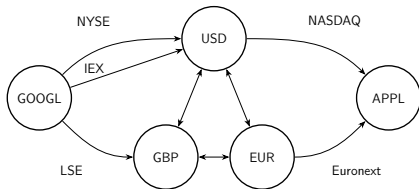
Nonlinear networks are everywhere!

- ▶ Power dispatch
- ▶ Communication systems
- ▶ Arrow-Debreu markets (market clearing)



Nonlinear networks are everywhere!

- ▶ Power dispatch
- ▶ Communication systems
- ▶ Arrow-Debreu markets (market clearing)
- ▶ Financial order routing



Nonlinear networks are everywhere!

- ▶ Power dispatch
- ▶ Communication systems
- ▶ Arrow-Debreu markets (market clearing)
- ▶ Financial order routing
- ▶ Supply chain with spoilage

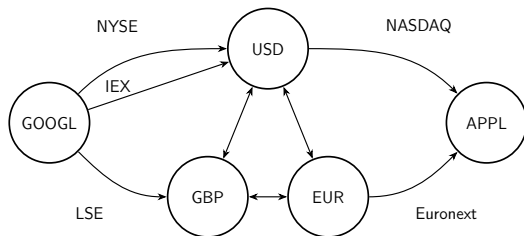
Nonlinear networks are everywhere!

- ▶ Power dispatch
- ▶ Communication systems
- ▶ Arrow-Debreu markets (market clearing)
- ▶ Financial order routing
- ▶ Supply chain with spoilage
- ▶ ...

Motivating example: order routing in financial networks

(Diamandis et al., Financial Cryptography 2023)

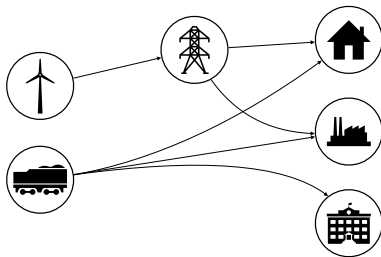
- ▶ **Goal:** convert fixed amount of Google stock to maximum amount of Apple stock
- ▶ Many venues on which you can place the order



- ▶ **Nonlinearity:** Price impact of trading: the more you trade, worse the price

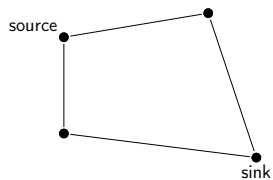
Motivating example: optimal power flow

- ▶ **Goal:** generate power to meet demand at minimum cost
- ▶ Multiple sources and transmission lines to choose from



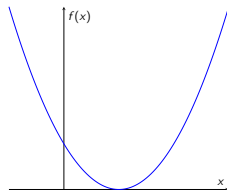
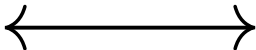
- ▶ **Nonlinearity:** Power loss: the more power transmitted, the more is dissipated

Perhaps a generalization?



Linear network flows

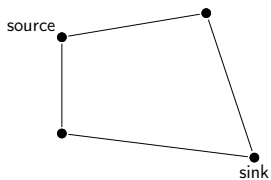
- ▶ Very fast to solve
- ▶ But not very expressive



Convex Optimization

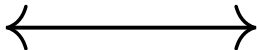
- ▶ Very expressive
- ▶ But slow in general

Perhaps a generalization?

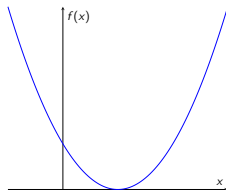


Linear network flows

- ▶ Very fast to solve
- ▶ But not very expressive



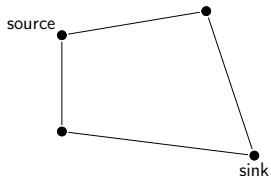
What do we do between?



Convex Optimization

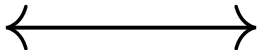
- ▶ Very expressive
- ▶ But slow in general

Perhaps a generalization?



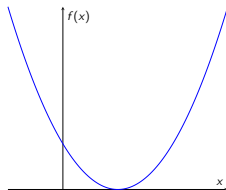
Linear network flows

- ▶ Very fast to solve
- ▶ But not very expressive



What do we do between?

A new framework:
Convex Network Flows



Convex Optimization

- ▶ Very expressive
- ▶ But slow in general

Outline

Motivation

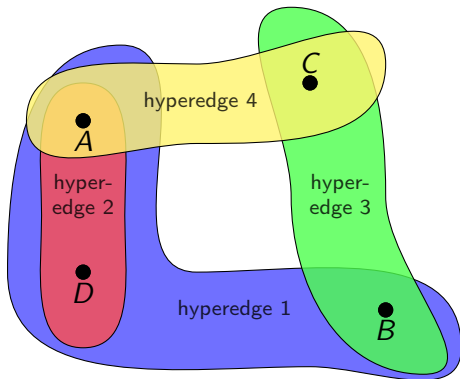
Framework

Applications

Algorithm

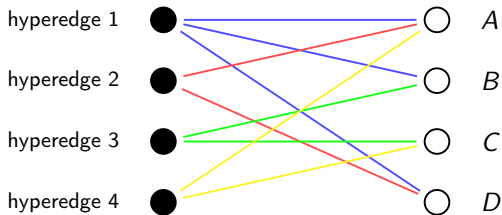
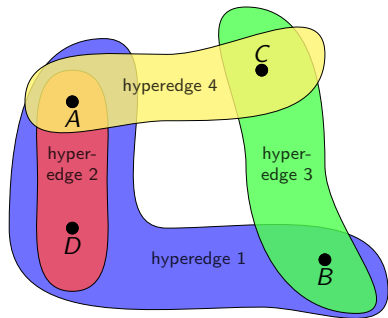
First generalization: hypergraphs

- Graph $\rightarrow$ **hypergraph**: edges can connect more than 2 vertices



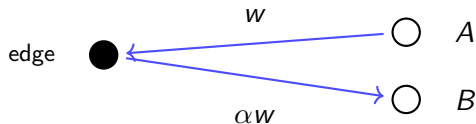
First generalization: hypergraphs

- Graph $\rightarrow$ **hypergraph**: edges can connect more than 2 vertices



Second generalization: convex sets

- Linear input-output relationship $\longrightarrow$ convex set of *allowable flows* T

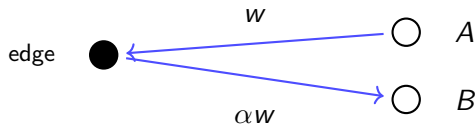


Input-output relationship: $h(w) = \alpha w$

$$\text{edge flow } z = \begin{bmatrix} -w \\ h(w) \end{bmatrix}$$

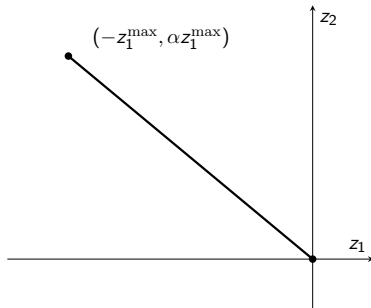
Second generalization: convex sets

- Linear input-output relationship $\longrightarrow$ convex set of *allowable flows* T



Input-output relationship: $h(w) = \alpha w$

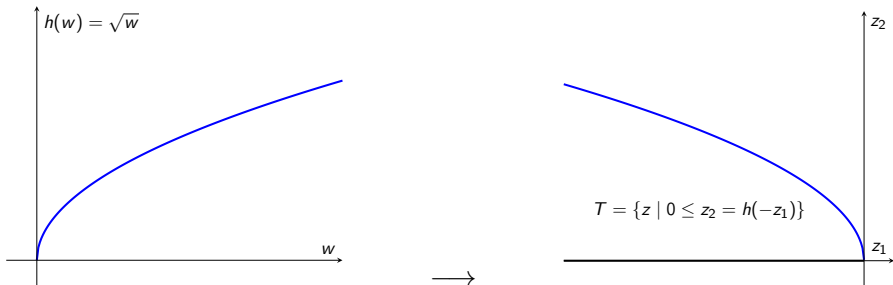
$$\text{edge flow } z = \begin{bmatrix} -w \\ h(w) \end{bmatrix}$$



$$T = \{z \in \mathbb{R}^2 \mid z_2 = \alpha \cdot (-z_1)\}$$

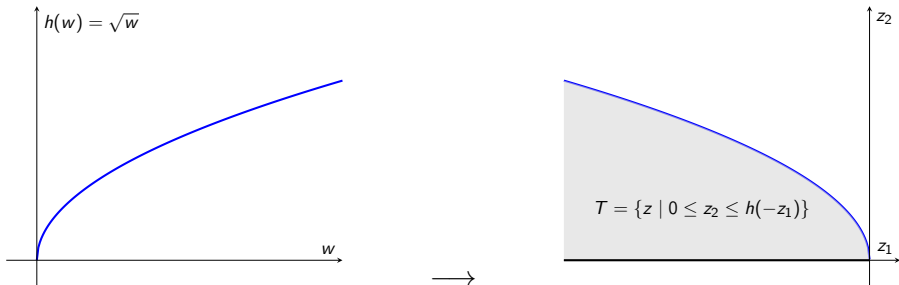
Second generalization: convex sets

- Linear input-output relationship $\longrightarrow$ convex set



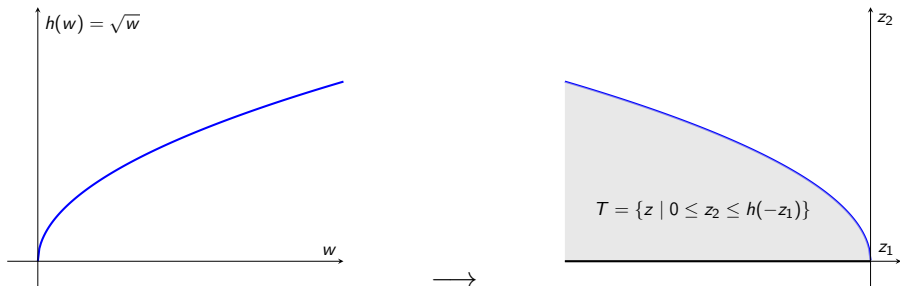
Second generalization: convex sets

- Linear input-output relationship $\longrightarrow$ convex set



Second generalization: convex sets

- ▶ Linear input-output relationship $\longrightarrow$ convex set



- ▶ Claim: we can ensure a solution flow always lies on the boundary (more soon)

Accounting

- ▶ Net flow $y \in \mathbb{R}^n$ and edge flows $x_i \in T_i \subseteq \mathbb{R}^{n_i}$

Accounting

- ▶ Net flow $y \in \mathbb{R}^n$ and edge flows $x_i \in T_i \subseteq \mathbb{R}^{n_i}$
- ▶ Matrices A_i map flow's *local* index in edge i to *global* index, e.g.,

node	Local Index	Global Index
B	1	2
D	2	4

$$A_i \cdot \begin{bmatrix} -0.5 \\ 0.3 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.5 \\ 0 \\ 0.3 \\ \vdots \end{bmatrix} \begin{matrix} A \\ B \\ C \\ D \end{matrix}$$

Accounting

- ▶ Net flow $y \in \mathbb{R}^n$ and edge flows $x_i \in T_i \subseteq \mathbb{R}^{n_i}$
- ▶ Matrices A_i map flow's *local* index in edge i to *global* index, e.g.,

node	Local Index	Global Index
B	1	2
D	2	4

$$A_i \cdot \begin{bmatrix} -0.5 \\ 0.3 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.5 \\ 0 \\ 0.3 \\ \vdots \end{bmatrix} \begin{matrix} A \\ B \\ C \\ D \end{matrix}$$

- ▶ The overall *net flow* in the network is

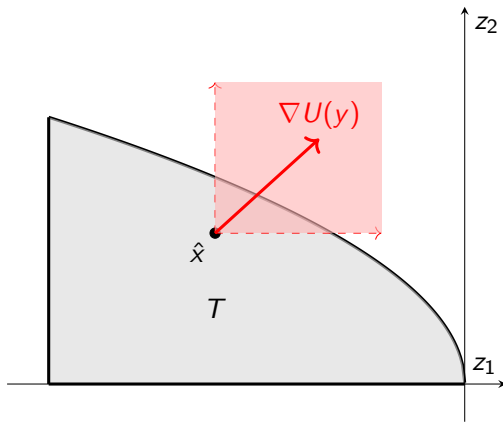
$$y = \sum_{i=1}^m A_i x_i$$

Objective: maximize utility

- Concave, increasing utility functions for net flow $U(y)$ and edge flows $\{V_i(x_i)\}$

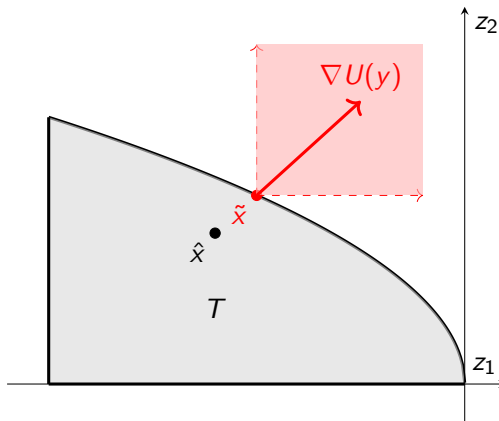
Objective: maximize utility

- Concave, increasing utility functions for net flow $U(y)$ and edge flows $\{V_i(x_i)\}$



Objective: maximize utility

- Concave, increasing utility functions for net flow $U(y)$ and edge flows $\{V_i(x_i)\}$



Framework: convex flow problem

(Diamandis et al., arXiv preprint 2024)

► The *convex flow problem*:

$$\begin{aligned} &\text{maximize} && U(y) + \sum_{i=1}^m V_i(x_i) \\ &\text{subject to} && y = \sum_{i=1}^m A_i x_i \\ &&& x_i \in T_i, \quad i = 1, \dots, m. \end{aligned}$$

Framework: convex flow problem

(Diamandis et al., arXiv preprint 2024)

- The *convex flow problem*:

$$\begin{aligned} \text{maximize} \quad & U(y) + \sum_{i=1}^m V_i(x_i) \\ \text{subject to} \quad & y = \sum_{i=1}^m A_i x_i \\ & x_i \in T_i, \quad i = 1, \dots, m. \end{aligned}$$

- Maximize utility of net flow plus utility of edge flows

Framework: convex flow problem

(Diamandis et al., arXiv preprint 2024)

- The *convex flow problem*:

$$\begin{aligned} &\text{maximize} && U(y) + \sum_{i=1}^m V_i(x_i) \\ &\text{subject to} && y = \sum_{i=1}^m A_i x_i \\ &&& x_i \in T_i, \quad i = 1, \dots, m. \end{aligned}$$

- Maximize utility of net flow plus utility of edge flows
- Subject to having flow conservation and allowable edge flows

Framework: convex flow problem

(Diamandis et al., arXiv preprint 2024)

- ▶ The *convex flow problem*:

$$\begin{aligned} &\text{maximize} && U(y) + \sum_{i=1}^m V_i(x_i) \\ &\text{subject to} && y = \sum_{i=1}^m A_i x_i \\ &&& x_i \in T_i, \quad i = 1, \dots, m. \end{aligned}$$

- ▶ Maximize utility of net flow plus utility of edge flows
- ▶ Subject to having flow conservation and allowable edge flows
- ▶ **Aside:** definition of allowable flows T_i 's allows for a DCP-like 'calculus' of flows

Outline

Motivation

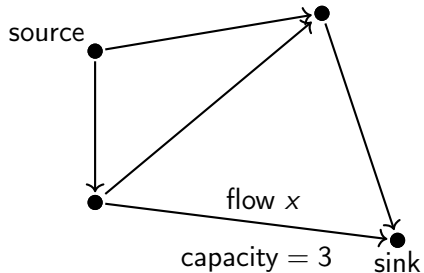
Framework

Applications

Algorithm

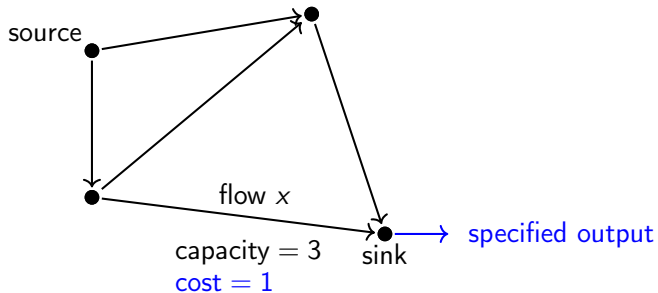
Maximum flow & friends

► Maximum flow problem



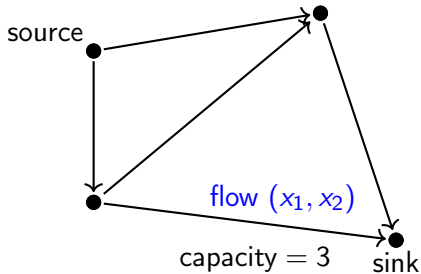
Maximum flow & friends

- ▶ Maximum flow problem
- ▶ Minimum cost flow



Maximum flow & friends

- ▶ Maximum flow problem
- ▶ Minimum cost flow
- ▶ Multi-commodity flows



Maximum flow & friends

- ▶ Maximum flow problem
- ▶ Minimum cost flow
- ▶ Multi-commodity flows
- ▶ and generalizations...

Application: optimal power flow

(from Stursberg 2019)

- ▶ Goal: find cost-minimizing power generation plan that satisfies demand

Application: optimal power flow

(from Stursberg 2019)

- ▶ Goal: find cost-minimizing power generation plan that satisfies demand
- ▶ **Graph:** transmission line network



Image source: Lawrence Berkeley National Laboratory

Application: optimal power flow

(from Stursberg 2019)

- ▶ Goal: find cost-minimizing power generation plan that satisfies demand
- ▶ **Graph:** transmission line network
- ▶ **Flow:** power



Image source: Lawrence Berkeley National Laboratory

Application: optimal power flow

(from Stursberg 2019)

- ▶ Goal: find cost-minimizing power generation plan that satisfies demand
- ▶ **Graph:** transmission line network
- ▶ **Flow:** power
- ▶ **Allowable flows:** transmission line physics

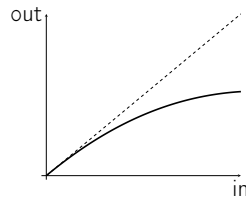


Image source: Lawrence Berkeley National Laboratory

Application: optimal power flow

(from Stursberg 2019)

- ▶ Goal: find cost-minimizing power generation plan that satisfies demand
- ▶ **Graph:** transmission line network
- ▶ **Flow:** power
- ▶ **Allowable flows:** transmission line physics
- ▶ **Objective:** minimize cost of power generation

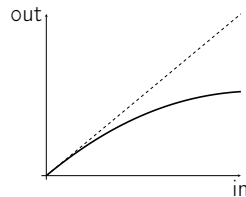
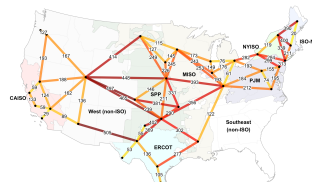


Image source: Lawrence Berkeley National Laboratory

Application: routing in wireless networks

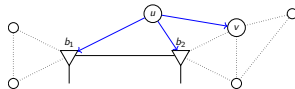
(from Xiao et al. 2004)

- ▶ Goal: maximize data rate, subject to power and bandwidth constraints

Application: routing in wireless networks

(from Xiao et al. 2004)

- ▶ Goal: maximize data rate, subject to power and bandwidth constraints
- ▶ **Hypergraph:** communication network

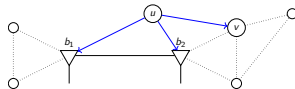


Hyperedges: all outgoing neighbors

Application: routing in wireless networks

(from Xiao et al. 2004)

- ▶ Goal: maximize data rate, subject to power and bandwidth constraints
- ▶ **Hypergraph:** communication network
- ▶ **Flow:** data rate

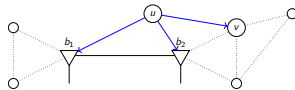


Hyperedges: all outgoing neighbors

Application: routing in wireless networks

(from Xiao et al. 2004)

- ▶ Goal: maximize data rate, subject to power and bandwidth constraints
- ▶ **Hypergraph:** communication network
- ▶ **Flow:** data rate
- ▶ **Allowable flows:** channel capacity (multicast)
with total power & bandwidth constraints

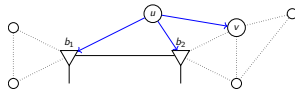


Hyperedges: all outgoing neighbors

Application: routing in wireless networks

(from Xiao et al. 2004)

- ▶ Goal: maximize data rate, subject to power and bandwidth constraints
- ▶ **Hypergraph:** communication network
- ▶ **Flow:** data rate
- ▶ **Allowable flows:** channel capacity (multicast) with total power & bandwidth constraints
- ▶ **Objective:** maximize data rate from source to destination



Hyperedges: all outgoing neighbors

Application: market clearing

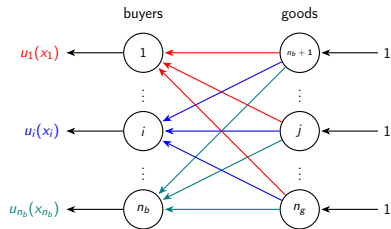
(from Végh 2014)

- ▶ Goal: find market clearing prices for to allocate divisible goods to buyers

Application: market clearing

(from Véghe 2014)

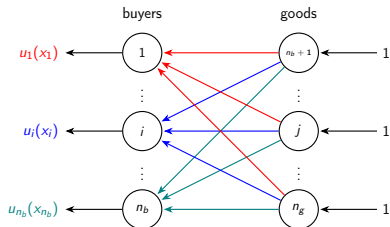
- Goal: find market clearing prices for to allocate divisible goods to buyers
- **Hypergraph:** each buyer linked to all goods



Application: market clearing

(from Véghe 2014)

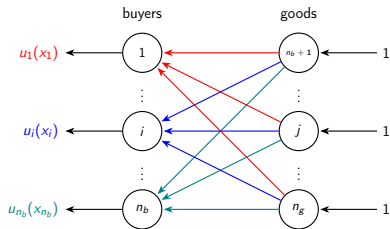
- ▶ Goal: find market clearing prices for to allocate divisible goods to buyers
- ▶ **Hypergraph:** each buyer linked to all goods
- ▶ **Flow:** goods & utility



Application: market clearing

(from Véggh 2014)

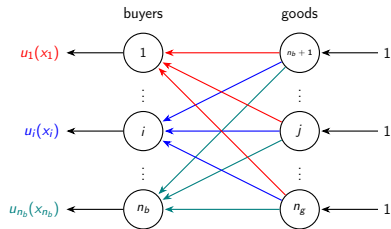
- ▶ Goal: find market clearing prices for to allocate divisible goods to buyers
- ▶ **Hypergraph:** each buyer linked to all goods
- ▶ **Flow:** goods & utility
- ▶ **Allowable flows:** amount of goods, nonlinear utility of buyers (can include complements)



Application: market clearing

(from Véghe 2014)

- ▶ Goal: find market clearing prices for to allocate divisible goods to buyers
- ▶ **Hypergraph:** each buyer linked to all goods
- ▶ **Flow:** goods & utility
- ▶ **Allowable flows:** amount of goods, nonlinear utility of buyers (can include complements)
- ▶ **Objective:** maximize sum of log utilities



Application: optimal orders in asset networks

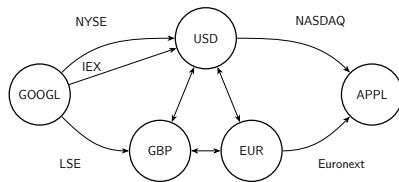
(from Diamandis et al. 2023)

- ▶ Goal: maximize price-weighted output for fixed input portfolio

Application: optimal orders in asset networks

(from Diamandis et al. 2023)

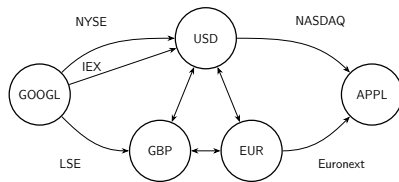
- ▶ Goal: maximize price-weighted output for fixed input portfolio
- ▶ **Hypergraph:** assets (nodes) linked by markets (edges)



Application: optimal orders in asset networks

(from Diamandis et al. 2023)

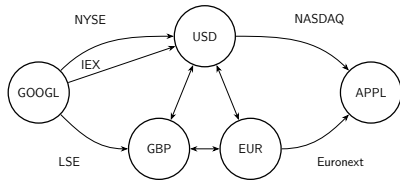
- ▶ Goal: maximize price-weighted output for fixed input portfolio
- ▶ **Hypergraph:** assets (nodes) linked by markets (edges)
- ▶ **Flow:** assets



Application: optimal orders in asset networks

(from Diamandis et al. 2023)

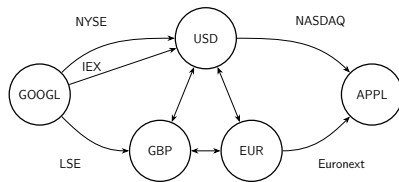
- ▶ Goal: maximize price-weighted output for fixed input portfolio
- ▶ **Hypergraph:** assets (nodes) linked by markets (edges)
- ▶ **Flow:** assets
- ▶ **Allowable flows:** market structure



Application: optimal orders in asset networks

(from Diamandis et al. 2023)

- ▶ Goal: maximize price-weighted output for fixed input portfolio
- ▶ **Hypergraph:** assets (nodes) linked by markets (edges)
- ▶ **Flow:** assets
- ▶ **Allowable flows:** market structure
- ▶ **Objective:** maximize output



And a lot more...

- ▶ Queueing networks (Bertsekas et al. 1992, §5.4)
- ▶ Routing games (Roughgarden 2007, §18)
- ▶ Supply chains with spoilage (Nagurney et al. 2022, §2.3)
- ▶ Reservoir network management (Bertsekas 1998, §8.1)
- ▶ Allocating computing resources (Agrawal et al. 2022)
- ▶ Supply chain allocation problems (Schütz et al. 2009)
- ▶ Wireless network resource allocation (Chiang et al. 2007)

Outline

Motivation

Framework

Applications

Algorithm

Dual problem

► Dual problem:

$$\text{minimize } g(\nu, \eta) = \bar{U}(\nu) + \sum_{i=1}^m \left(\bar{V}_i(\eta_i - A_i^T \nu) + f_i(\eta_i) \right),$$

where

$$\bar{U}(\nu) = \sup_y (U(y) - \nu^T y),$$

$$\bar{V}_i(\xi) = \sup_{x_i} (V_i(x_i) - \xi^T x_i),$$

$$f_i(\tilde{\eta}) = \sup_{\tilde{x}_i \in T_i} \tilde{\eta}^T \tilde{x}_i.$$

Subproblems

- ▶ Maximum utility problem:

$$\bar{U}(\nu) = \sup_y (U(y) - \nu^T y) = (-U)^*(-\nu)$$

Subproblems

- ▶ Maximum utility problem:

$$\bar{U}(\nu) = \sup_y (U(y) - \nu^T y) = (-U)^*(-\nu)$$

- ▶ Arbitrage problem:

$$f_i(\eta) = \sup_{\tilde{x}_i \in T_i} \eta^T \tilde{x}_i$$

Subproblems

- ▶ Maximum utility problem:

$$\bar{U}(\nu) = \sup_y (U(y) - \nu^T y) = (-U)^*(-\nu)$$

- ▶ Arbitrage problem:

$$f_i(\eta) = \sup_{\tilde{x}_i \in T_i} \eta^T \tilde{x}_i$$

Example: $w \mapsto h(w)$

$$f_i(\eta) = \sup_w \{\eta_2 h(w) - \eta_1 w\}$$

Subproblems

- ▶ Maximum utility problem:

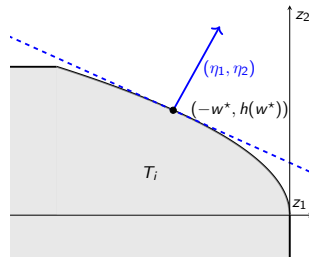
$$\bar{U}(\nu) = \sup_y (U(y) - \nu^T y) = (-U)^*(-\nu)$$

- ▶ Arbitrage problem:

$$f_i(\eta) = \sup_{\tilde{x}_i \in T_i} \eta^T \tilde{x}_i$$

Example: $w \mapsto h(w)$

$$f_i(\eta) = \sup_w \{ \eta_2 h(w) - \eta_1 w \}$$



Dual variables as prices

► Optimality conditions:

$$\nabla U(y^*) = \nu^*,$$

$$\nabla V_i(x_i^*) = \eta_i^* - A_i^T \nu^*, \quad i = 1, \dots, m$$

$$\eta_i^* \in \mathcal{N}_i(\tilde{x}_i^*), \quad i = 1, \dots, m.$$

'global' net flow prices

'local' edge flow prices

no arbitrage

Dual variables as prices

- Optimality conditions:

$$\nabla U(y^*) = \nu^*,$$

$$\nabla V_i(x_i^*) = \eta_i^* - A_i^T \nu^*, \quad i = 1, \dots, m$$

$$\eta_i^* \in \mathcal{N}_i(\tilde{x}_i^*), \quad i = 1, \dots, m.$$

‘global’ net flow prices

‘local’ edge flow prices

no arbitrage

- Special case: $V_i = 0$:

$$\nabla U(y^*) = \nu^*$$

$$A_i^T \nu^* \in \mathcal{N}_i(x_i^*)$$

Dual variables as prices

► Optimality conditions:

$$\nabla U(y^*) = \nu^*,$$

$$\nabla V_i(x_i^*) = \eta_i^* - A_i^T \nu^*, \quad i = 1, \dots, m$$

$$\eta_i^* \in \mathcal{N}_i(\tilde{x}_i^*), \quad i = 1, \dots, m.$$

'global' net flow prices

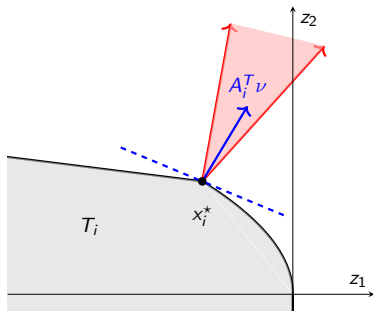
'local' edge flow prices

no arbitrage

► Special case: $V_i = 0$:

$$\nabla U(y^*) = \nu^*$$

$$A_i^T \nu^* \in \mathcal{N}_i(x_i^*)$$



Evaluating the dual function

- ▶ 'Conjugate functions' $\bar{U}$ and $\bar{V}_i$ are easy, solution y^*, x_i^*

Evaluating the dual function

- ▶ 'Conjugate functions' $\bar{U}$ and $\bar{V}_i$ are easy, solution y^*, x_i^*
- ▶ 'Arbitrage problem' f_i : closed form or small root finding problem, solution $\tilde{x}_i^*$

Evaluating the dual function

- ▶ 'Conjugate functions' $\bar{U}$ and $\bar{V}_i$ are easy, solution y^*, x_i^*
- ▶ 'Arbitrage problem' f_i : closed form or small root finding problem, solution $\tilde{x}_i^*$
 - Parallelized across all edges

Evaluating the dual function

- ▶ 'Conjugate functions' $\bar{U}$ and $\bar{V}_i$ are easy, solution y^*, x_i^*
- ▶ 'Arbitrage problem' f_i : closed form or small root finding problem, solution $\tilde{x}_i^*$
 - Parallelized across all edges
- ▶ Together, give us $g(\nu, \eta)$ and gradient:

$$\begin{aligned}\nabla_{\nu} g(\nu^*, \eta^*) &= y^* - \sum_{i=1}^m A_i x_i^*, \\ \nabla_{\eta_i} g(\nu^*, \eta^*) &= x_i^* - \tilde{x}_i^*, \quad i = 1, \dots, m.\end{aligned}$$

Evaluating the dual function

- ▶ ‘Conjugate functions’ $\bar{U}$ and $\bar{V}_i$ are easy, solution y^*, x_i^*
- ▶ ‘Arbitrage problem’ f_i : closed form or small root finding problem, solution $\tilde{x}_i^*$
 - Parallelized across all edges
- ▶ Together, give us $g(\nu, \eta)$ and gradient:

$$\begin{aligned}\nabla_{\nu} g(\nu^*, \eta^*) &= y^* - \sum_{i=1}^m A_i x_i^*, \\ \nabla_{\eta_i} g(\nu^*, \eta^*) &= x_i^* - \tilde{x}_i^*, \quad i = 1, \dots, m.\end{aligned}$$

- ▶ Can solve dual problem with any first-order method

Evaluating the dual function

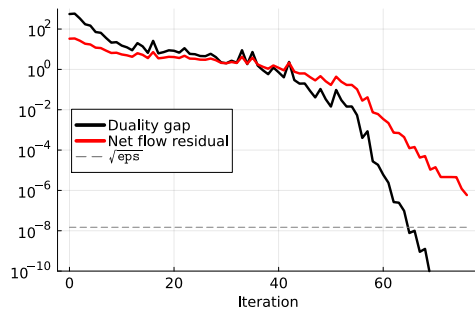
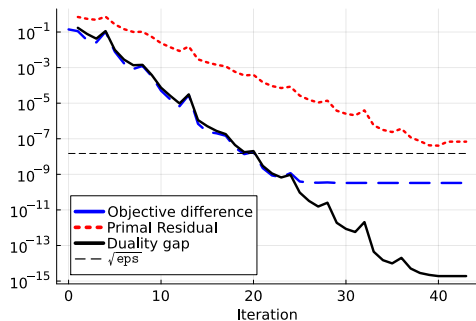
- ▶ 'Conjugate functions' $\bar{U}$ and $\bar{V}_i$ are easy, solution y^*, x_i^*
- ▶ 'Arbitrage problem' f_i : closed form or small root finding problem, solution $\tilde{x}_i^*$
 - Parallelized across all edges
- ▶ Together, give us $g(\nu, \eta)$ and gradient:

$$\begin{aligned}\nabla_{\nu} g(\nu^*, \eta^*) &= y^* - \sum_{i=1}^m A_i x_i^*, \\ \nabla_{\eta_i} g(\nu^*, \eta^*) &= x_i^* - \tilde{x}_i^*, \quad i = 1, \dots, m.\end{aligned}$$

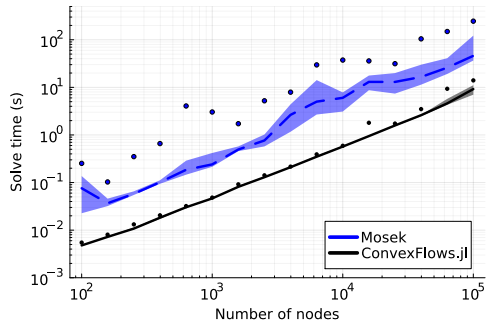
- ▶ Can solve dual problem with any first-order method
- ▶ If not strictly convex, simple method to recover feasible solution

Example: optimal power flow

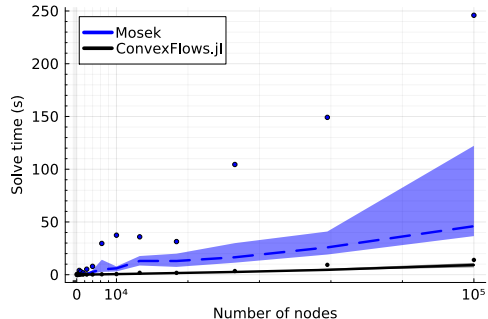
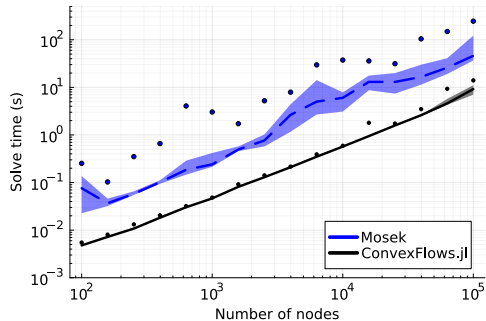
- Convergence with L-BFGS-B, after change of variables, (left) and BFGS (right)



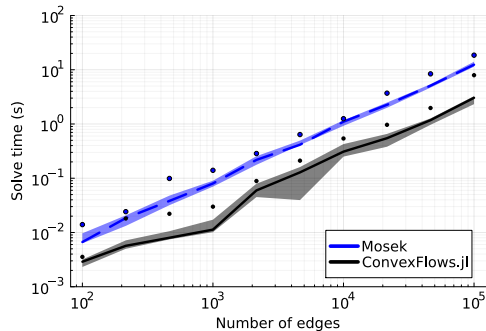
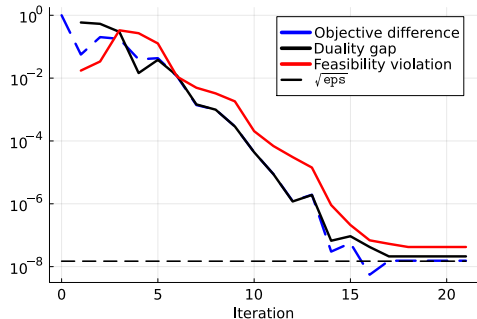
Example: optimal power flow



Example: optimal power flow



Example: financial order routing



Conjugates? Support functions?

Conjugates? Support functions?

How does a 'normal' user specify a problem?

A Simple Interface: `ConvexFlows.jl`

(from Diamandis et al. 2024a)

- ▶ Problem specification: library of objective functions, specify edge gain functions:

```
h(w) = 3w - 16.0*(log(1 + exp(0.25 * w)) - log(2))  
push!(edges, Edge((i, j); h=h, ub=3.0))
```

- ▶ CVX-like ability to naturally specify problems and quickly test them
- ▶ Specify subproblems using `JuMP.jl`
- ▶ Performance-sensitive users: specify subproblem solutions directly

Full problem specification

```
# Parameters: demand d, graph Adj, upper bounds ub

obj = NonpositiveQuadratic(d)

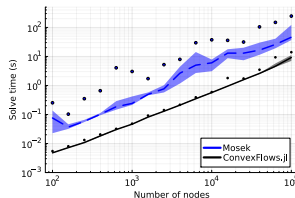
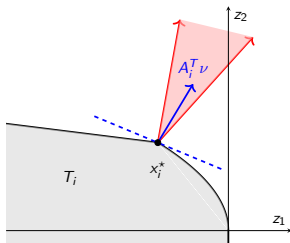
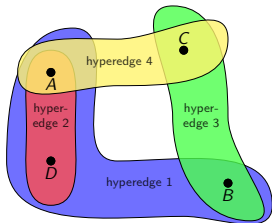
h(w) = 3w - 16.0*(log1pexp(0.25 * w) - log(2))

lines = Edge[]
for i in 1:n, j in i+1:n
    Adj[i, j] ≤ 0 && continue
    push!(lines, Edge((i, j); h=h, ub=ub[i]))
end

prob = problem(obj=obj, edges=lines)
result = solve!(prob)
```

Questions? (email: tdiamand@mit.edu)

- ▶ New framework for **nonlinear** network flows over **hypergraphs**
 - Generalizes many classic results & includes many problems in the literature
- ▶ Fast algorithm to solve, implemented in `ConvexFlows.jl`
- ▶ Natural fixed fee and decentralized extensions to model and algorithm



Appendix

Appendix

A bit of history

More theory

- Flow calculus

- Nonconvex flows

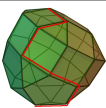
- Theory Miscellanea

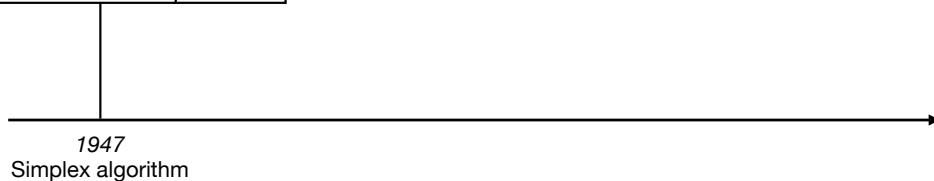
Motivating example details

Numerical experiment details

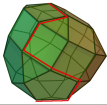
Additional algorithmic considerations

Works cited

$\begin{array}{ll}\text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \succeq 0.\end{array}$	
---	---

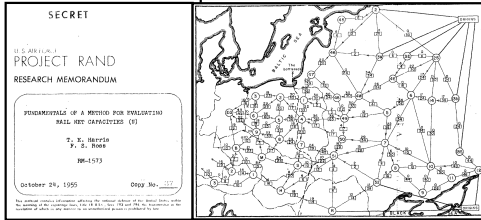


minimize $c^T x$
 subject to $Ax = b$
 $x \succeq 0$.

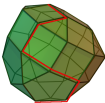


Max flow & min cut
 1954

1947
 Simplex algorithm



minimize $c^T x$
subject to $Ax = b$
 $x \succeq 0$.



MAXIMAL FLOW THROUGH A NETWORK

L. R. FORD, JR. AND D. R. FULKERSON

Max flow & min cut
1954

1947
Simplex algorithm

1955
Ford-Fulkerson algorithm

SECRET

O S AIRTEL 3

PROJECT RAND

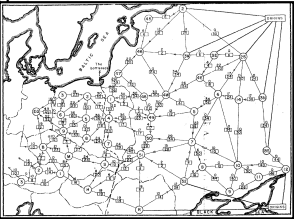
RESEARCH MEMORANDUM

FUNDAMENTALS OF A METHOD FOR EVALUATING
RAIL NET CAPACITIES (S)

T. K. Harris
F. S. Ross
RM-1573

October 24, 1955

Copy No. 227



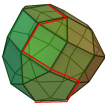
A bit of history

30

minimize $c^T x$

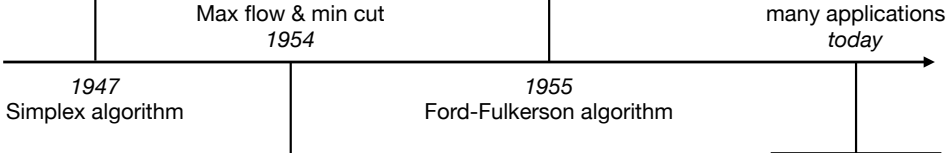
subject to $Ax = b$

$x \succeq 0.$



MAXIMAL FLOW THROUGH A NETWORK

L. R. FORD, JR. AND D. R. FULKERSON



SECRET

U.S. AIR FORCE

PROJECT RAND

RESEARCH MEMORANDUM

FUNDAMENTALS OF A METHOD FOR EVALUATING

RAIL NET CAPACITIES (S)

T. K. Harris

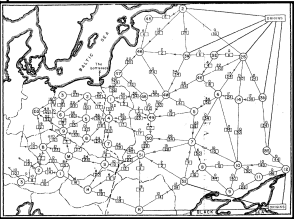
F. S. Ross

RM-1573

October 24, 1955

Copy No. 227

This material contains information affecting the national defense of the United States within the meaning of the espionage laws, Title 18, U.S.C., Sec. 793 and 794, and the transmission or the revelation of its contents in any manner to an unauthorized person is prohibited by law.



NETWORK FLOWS

THEORY, ALGORITHMS, AND APPLICATIONS

RAVENDRA K. AHUJA

THOMAS L. MAGNANTI

JAMES B. ORLIN

Appendix

A bit of history

More theory

- Flow calculus

- Nonconvex flows

- Theory Miscellanea

Motivating example details

Numerical experiment details

Additional algorithmic considerations

Works cited

Appendix

A bit of history

More theory

- Flow calculus

- Nonconvex flows

- Theory Miscellanea

Motivating example details

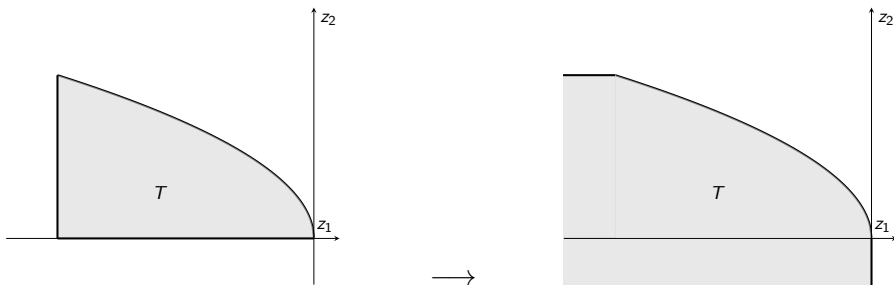
Numerical experiment details

Additional algorithmic considerations

Works cited

Downward closure

- Idea: allow any 'worse' point



Sets of allowable flows

(inspired by Angeris et al. 2023)

- ▶ A set of *allowable flows* $T \subseteq \mathbb{R}^n$ must
 - be closed and convex
 - be downward closed
 - contain the zero vector

Sets of allowable flows

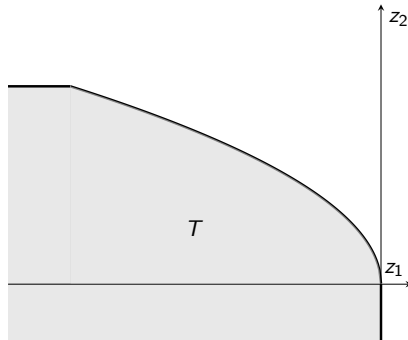
(inspired by Angeris et al. 2023)

- ▶ A set of *allowable flows* $T \subseteq \mathbb{R}^n$ must
 - be closed and convex
 - be downward closed
 - contain the zero vector
 - be bounded: $b - T \subseteq \mathbb{R}_+^n$ for some b

Sets of allowable flows

(inspired by Angeris et al. 2023)

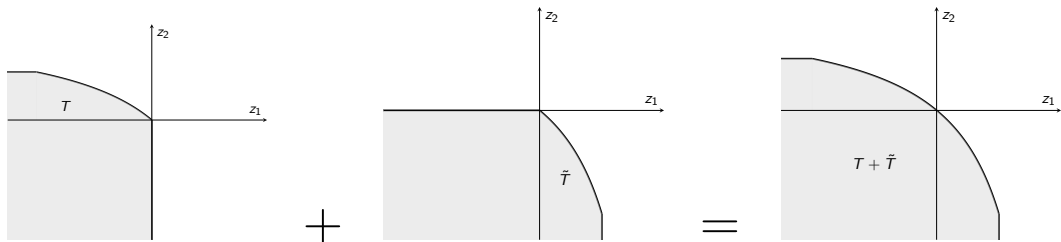
- ▶ A set of *allowable flows* $T \subseteq \mathbb{R}^n$ must
 - be closed and convex
 - be downward closed
 - contain the zero vector
 - be bounded: $b - T \subseteq \mathbb{R}_+^n$ for some b



Flow calculus

We can combine, split, and transform sets of allowable flows:

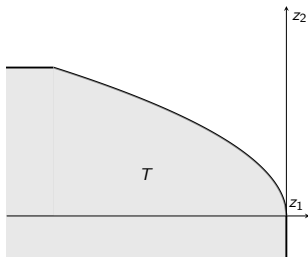
► Addition (Minkowski)



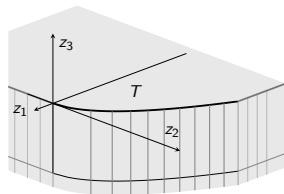
Flow calculus

We can combine, split, and transform sets of allowable flows:

- ▶ Addition (Minkowski)
- ▶ Scaling by a nonnegative injective matrix: $T_i \longrightarrow AT_i - \mathbb{R}^n$ (e.g., lifting)

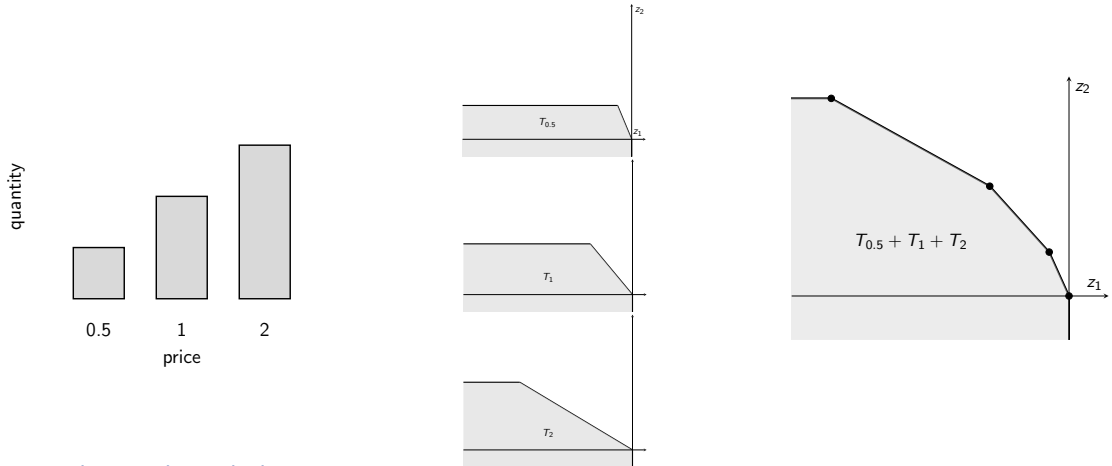


$$\begin{bmatrix} 1 & \\ & 1 \end{bmatrix} - \mathbb{R}^3 \longrightarrow$$



Aggregate edge example

- Orderbook markets: many linear edges or one piecewise linear edge



Appendix

A bit of history

More theory

Flow calculus

Nonconvex flows

Theory Miscellanea

Motivating example details

Numerical experiment details

Additional algorithmic considerations

Works cited

Fixed costs

(from Diamandis et al. 2024b)

- Natural extension: **fixed cost** q_i to use edge i

$$\begin{aligned} \text{maximize} \quad & U(y) + \sum_{i=1}^m V_i(x_i) + q^T \lambda \\ \text{subject to} \quad & y = \sum_{i=1}^m A_i x_i \\ & (x_i, \lambda_i) \in \{(0, 0)\} \cup (T_i \times \{-1\}), \quad i = 1, \dots, m, \end{aligned}$$

Fixed costs

(from Diamandis et al. 2024b)

- Natural extension: **fixed cost** q_i to use edge i

$$\begin{aligned} &\text{maximize} && U(y) + \sum_{i=1}^m V_i(x_i) + q^T \lambda \\ &\text{subject to} && y = \sum_{i=1}^m A_i x_i \\ &&& (x_i, \lambda_i) \in \{(0, 0)\} \cup (T_i \times \{-1\}), \quad i = 1, \dots, m, \end{aligned}$$

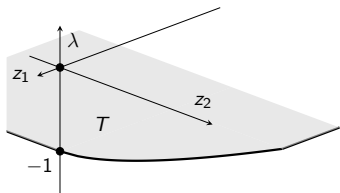
- Problem is nonconvex
- NP-hard to solve

Idea: convex relaxation

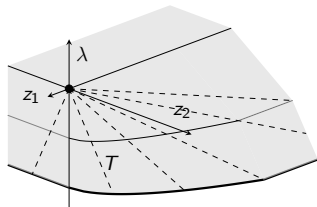
$$\begin{aligned} \text{maximize} \quad & U(y) + \sum_{i=1}^m V_i(x_i) + q^T \lambda \\ \text{subject to} \quad & y = \sum_{i=1}^m A_i x_i \\ & (x_i, \lambda_i) \in \text{conv}(\{(0, 0)\} \cup (T_i \times \{-1\})), \quad i = 1, \dots, m, \end{aligned}$$

Idea: convex relaxation

$$\begin{aligned} &\text{maximize} && U(y) + \sum_{i=1}^m V_i(x_i) + q^T \lambda \\ &\text{subject to} && y = \sum_{i=1}^m A_i x_i \\ &&& (x_i, \lambda_i) \in \text{conv}(\{(0, 0)\} \cup (T_i \times \{-1\})), \quad i = 1, \dots, m, \end{aligned}$$

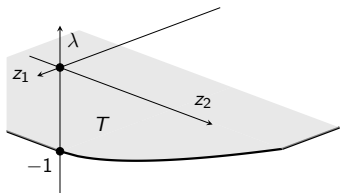


conv
→

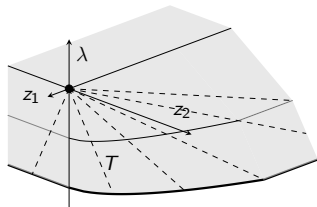


Idea: convex relaxation

$$\begin{aligned} & \text{maximize} && U(y) + \sum_{i=1}^m V_i(x_i) + q^T \lambda \\ & \text{subject to} && y = \sum_{i=1}^m A_i x_i \\ & && (x_i, \lambda_i) \in \text{conv}(\{(0, 0)\} \cup (T_i \times \{-1\})), \quad i = 1, \dots, m, \end{aligned}$$



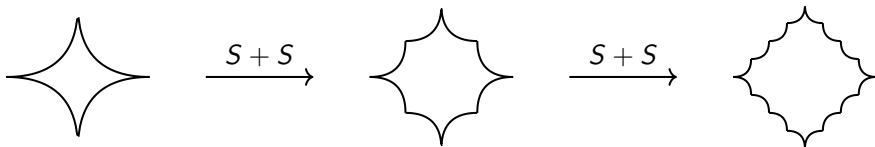
$\xrightarrow{\text{conv}}$



$$\text{conv}(\{(0, 0)\} \cup (T_i \times \{-1\})) = \text{cone}(\{(0, 0)\} \cup (T_i \times \{-1\})) \cap (\mathbb{R}^{n_i} \times [-1, 0]).$$

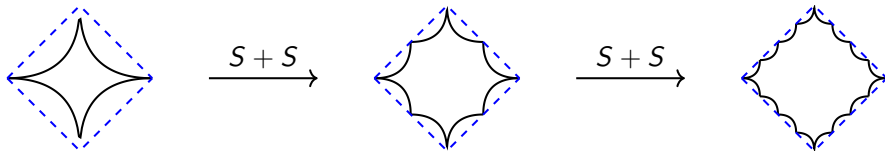
So what? Case with $V_i = 0$

- ▶ Often many more edges m than nodes n
- ▶ Shapley–Folkman Lemma: sum of many nonconvex sets is ‘almost’ convex



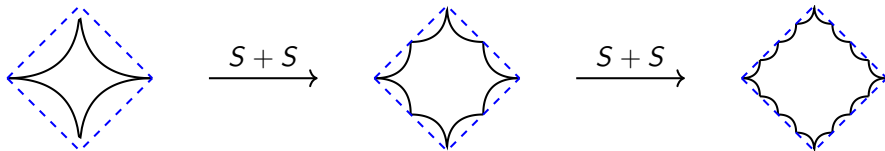
So what? Case with $V_i = 0$

- ▶ Often many more edges m than nodes n
- ▶ Shapley–Folkman Lemma: sum of many nonconvex sets is ‘almost’ convex



So what? Case with $V_i = 0$

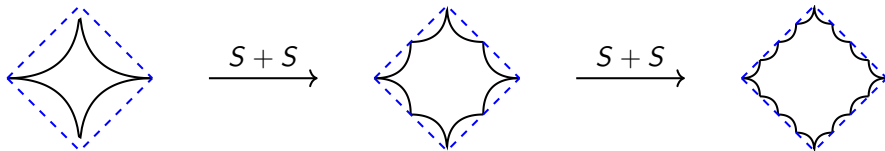
- ▶ Often many more edges m than nodes n
- ▶ Shapley–Folkman Lemma: sum of many nonconvex sets is ‘almost’ convex



- ▶ Can find optimal $\{(x_i, \lambda_i)\}$ with at most $n + 1$ non-integral λ_i 's

So what? Case with $V_i = 0$

- ▶ Often many more edges m than nodes n
- ▶ Shapley–Folkman Lemma: sum of many nonconvex sets is ‘almost’ convex



- ▶ Can find optimal $\{(x_i, \lambda_i)\}$ with at most $n + 1$ non-integral λ_i 's
- ▶ Optimal objective for relaxation p^0 satisfies $0 \leq p^0 - p^* \leq (n + 1)(\max_i q_i)$.

Structure in the dual problem suggests heuristic

- ▶ Recall that the dual problem requires solving the ‘arbitrage problem’, now

$$f_i^{\text{fees}}(\eta_i) = \max(f_i(\eta_i) - q_i, 0)$$

- ▶ Solution λ_i^* is 0 or -1 unless $f_i(\eta_i) = q_i$
- ▶ Easy to ‘almost’ solve convex flow problem with fixed fees

Structure in the dual problem suggests heuristic

- ▶ Recall that the dual problem requires solving the ‘arbitrage problem’, now

$$f_i^{\text{fees}}(\eta_i) = \max(f_i(\eta_i) - q_i, 0)$$

- ▶ Solution λ_i^* is 0 or -1 unless $f_i(\eta_i) = q_i$
- ▶ Easy to ‘almost’ solve convex flow problem with fixed fees
- ▶ But may be difficult to find feasible point y if constraints in U .

Structure in the dual problem suggests heuristic

- ▶ Recall that the dual problem requires solving the ‘arbitrage problem’, now

$$f_i^{\text{fees}}(\eta_i) = \max(f_i(\eta_i) - q_i, 0)$$

- ▶ Solution λ_i^* is 0 or -1 unless $f_i(\eta_i) = q_i$
- ▶ Easy to ‘almost’ solve convex flow problem with fixed fees
- ▶ But may be difficult to find feasible point y if constraints in U .
- ▶ Future work: conditions on when heuristic is exact

Appendix

A bit of history

More theory

Flow calculus

Nonconvex flows

Theory Miscellanea

Motivating example details

Numerical experiment details

Additional algorithmic considerations

Works cited

An augmenting path generalization

- ▶ Max Flow Problem: Flow is optimal iff there is no augmenting path
- ▶ Convex Flow Problem ($V_i = 0$): Flow is optimal iff there is no arbitrage

An augmenting path generalization

- ▶ Max Flow Problem: Flow is optimal iff there is no augmenting path
- ▶ Convex Flow Problem ($V_i = 0$): Flow is optimal iff there is no arbitrage
- ▶ Define

$$T_i^*(x_i) = \{\delta \mid x_i + t\delta_i \in T_i \text{ for some } t > 0\}.$$

- ▶ No arbitrage condition: for $\nu = \nabla U(y)$ and any $\delta_i \in T_i^*(x_i)$,

$$\nu^T \left(\sum_{i=1}^m A_i \delta_i \right) \leq 0.$$

NP-hard proof

- ▶ Knapsack problem: given $c \in \mathbb{Z}_+^n$ and $b \in \mathbb{Z}_+$, find $x \in \{0, 1\}^n$ such that $c^T x = b$.
- ▶ Reduction of knapsack (subset sum) problem to convex flow problem with fees:

$$\text{maximize } y - I(y \geq b) + c^T \lambda$$

$$\text{subject to } y = \sum_{i=1}^m A_i x_i$$

$$(x_i, \lambda_i) \in \{(0, 0)\} \cup ((-\infty, c_i] \times \{-1\}), \quad i = 1, \dots, m.$$

- ▶ Opt value 0 iff there exists solution to knapsack since

$$y + c^T \lambda = \sum_{i=1}^m (-\lambda_i) x_i + c^T \lambda \leq \sum_{i=1}^m c_i \lambda_i (-1 + 1) = 0$$

(Almost) self-dual problem

► Conic problem:

$$\begin{array}{ll}\text{maximize} & U(y) + \sum_{i=1}^m V_i(x_i) \\ \text{subject to} & y = \sum_{i=1}^m A_i x_i \\ & x_i \in K_i, \quad i = 1, \dots, m.\end{array}$$

► Dual problem:

$$\begin{array}{ll}\text{minimize} & \bar{U}(\nu) + \sum_{i=1}^m \bar{V}_i(\eta_i - \xi_i) \\ \text{subject to} & \nu = \sum_{i=1}^m A_i \xi_i \\ & \eta_i \in K_i^\circ, \quad i = 1, \dots, m.\end{array}$$

Appendix

A bit of history

More theory

- Flow calculus

- Nonconvex flows

- Theory Miscellanea

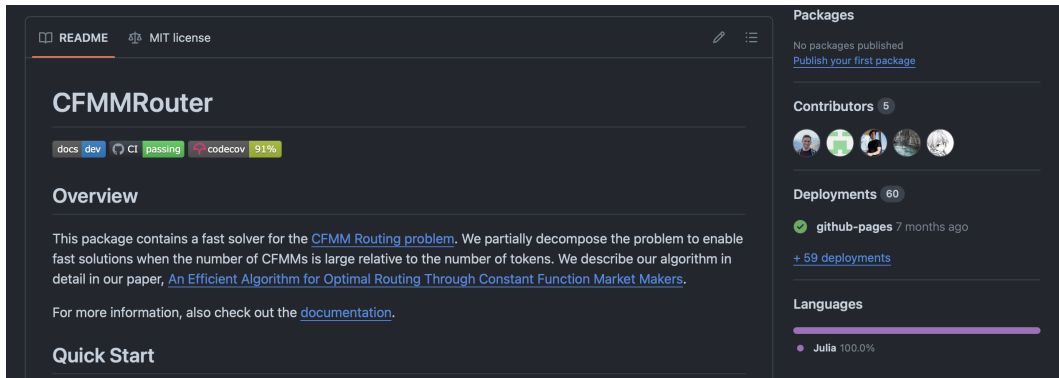
Motivating example details

Numerical experiment details

Additional algorithmic considerations

Works cited

My 2022 internship project



The screenshot shows the GitHub repository page for **CFMMRouter**. The repository is licensed under MIT and includes a README. The main content area displays the repository name, a status bar with 'docs', 'dev', 'CI', 'passing', 'codecov', and '91%', and an 'Overview' section. The overview text describes the package as a fast solver for the CFMM Routing problem, mentioning a partial decomposition and a reference to a paper. It also includes a 'Quick Start' section. On the right sidebar, there are sections for 'Packages' (no packages published), 'Contributors' (5 contributors), 'Deployments' (60 deployments, including github-pages 7 months ago), and 'Languages' (Julia 100.0%).

MIT license

CFMMRouter

docs dev CI passing codecov 91%

Overview

This package contains a fast solver for the [CFMM Routing problem](#). We partially decompose the problem to enable fast solutions when the number of CFMMs is large relative to the number of tokens. We describe our algorithm in detail in our paper, [An Efficient Algorithm for Optimal Routing Through Constant Function Market Makers](#).

For more information, also check out the [documentation](#).

Quick Start

Packages

No packages published
[Publish your first package](#)

Contributors 5

5 contributors

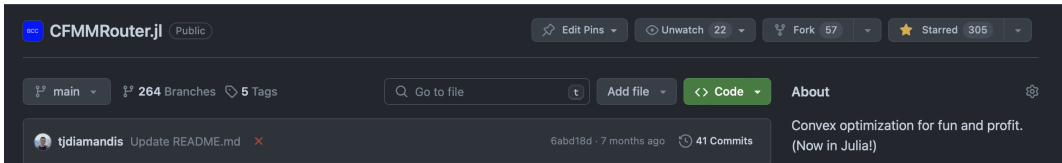
Deployments 60

✓ [github-pages](#) 7 months ago
[+ 59 deployments](#)

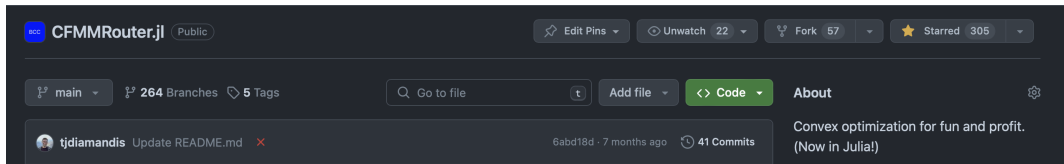
Languages

Julia 100.0%

It became quite popular...



It became quite popular...

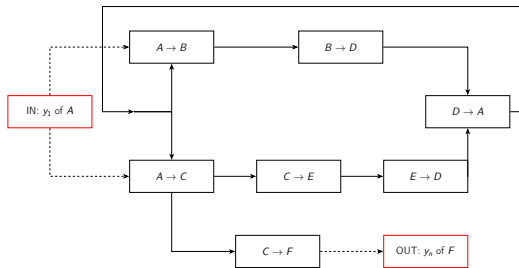


- ▶ A company (Flood) reached out to implement this (exact same algorithm)
- ▶ And I decided to write up a paper on it for Financial Cryptography

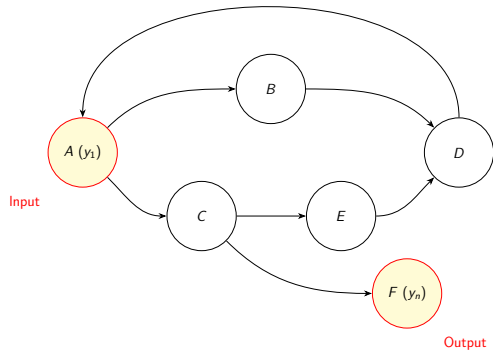
A real example

- Trade asset A for asset F

Markets as nodes:



Assets as nodes:



Motivating example details

A real example

- This one was executed in reality by Flood (flood.bid)

► From 0x41850fe2843b8... To 0xf111ed85e40931... For 23.760766 (\$23.86)  USD Coin (USDC)

► From DODO: USDT-US... To 0xf111ed85e40931... For 23.761929 (\$23.81)  Tether USD (USDT)

► From 0xf111ed85e40931... To DODO: USDT-US... For 23.780712 (\$23.88)  Bridged USDC (USDC.e)

► From DODO: USDT-US... To DODO: Multisig W... For 0.000237 (\$0.00)  Tether USD (USDT)

► From 0x7fcdc35463e37... To 0xf111ed85e40931... For 0.005066974652006983 (\$16.79)  Wrapped Ethe... (WETH)

► From 0xf111ed85e40931... To 0x7fcdc35463e37... For 17.162688 (\$17.23)  USD Coin (USDC)

► From 0xf111ed85e40931... To 0xc082398767ae7... For 2.817007 (\$2.83)  Bridged USDC (USDC.e)

► From 0xc082398767ae7... To 0xf111ed85e40931... For 0.00004204 (\$2.78)  Wrapped BTC (WBTC)

► From 0x562d29b54d2c5... To 0xf111ed85e40931... For 26.598204 (\$26.70)  Bridged USDC (USDC.e)

► From 0xf111ed85e40931... To 0x562d29b54d2c5... For 26.597903 (\$26.70)  USD Coin (USDC)

► From 0xf111ed85e40931... To 0x7050a8908e2a6... For 0.005066974652006984 (\$16.79)  Wrapped Ethe... (WETH)

► From 0x7050a8908e2a6... To 0xf111ed85e40931... For 0.953228882631324394 (\$17.17)  ChainLink To... (LINK)

► From 0x655c1607f8c2e... To 0xf111ed85e40931... For 20.000907 (\$20.08)  USD Coin (USDC)

► From 0xf111ed85e40931... To 0x655c1607f8c2e... For 1.109994504624138108 (\$19.99)  ChainLink To... (LINK)

► From 0xa79fd76ca2b24... To 0xf111ed85e40931... For 0.156826706956773007 (\$2.82)  ChainLink To... (LINK)

► From 0xf111ed85e40931... To 0xa79fd76ca2b24... For 0.00004204 (\$2.78)  Wrapped BTC (WBTC)

► From 0xf111ed85e40931... To 0x41850fe2843b8... For 23.761929 (\$23.81)  Tether USD (USDT)

Appendix

A bit of history

More theory

- Flow calculus

- Nonconvex flows

- Theory Miscellanea

Motivating example details

Numerical experiment details

Additional algorithmic considerations

Works cited

Optimal power flow

- ▶ Goal: minimize a quadratic power generation cost for random demand

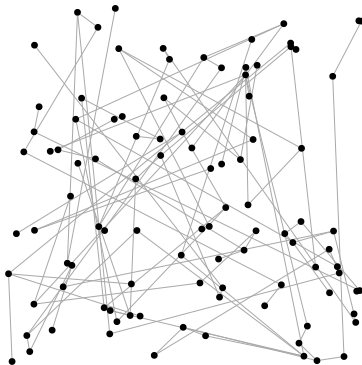
$$c_i(w) = \begin{cases} (1/2)w^2 & w \geq 0 \\ 0 & w < 0. \end{cases}$$

- ▶ Demand randomly sampled from $\{0.5, 1, 2\}$
- ▶ Power each node needs to generate is $d - y$, so objective is

$$U(y) = \sum_{i=1}^n -c_i(d_i - y_i).$$

Optimal power flow: network

- We generate the network as in Kraning et al. 2013



Optimal power flow: edges

- ▶ Edges have random capacity, sampled from $\{1, 2, 3\}$

- ▶ Power lost is

$$\ell(w) = 16 (\log(1 + \exp(w/4)) - \log 2) - 2w$$

- ▶ The set of allowable flows is

$$\mathcal{T} = \{z \in \mathbb{R}^2 \mid -b \leq z_1 \leq 0, z_2 \leq -z_1 - \ell(-z_1)\}$$

- ▶ Given prices η , the optimal input has a closed form:

$$x_1^* = - \left(4 \log \left(\frac{3\eta_2 - \eta_1}{\eta_2 + \eta_1} \right) \right)_{[0,b]}, \quad x_2^* = -x_1^* - \ell(-x_1^*)$$

Optimal power flow: conic form

- Compare the convex flow problem with the equivalent conic form

$$\begin{aligned} & \text{maximize} && -1^T t_1 \\ & \text{subject to} && (0.5, (t_1)_i, (t_2)_i) \in K_{\text{rot}2}, \quad \text{for } i = 1, \dots, n \\ & && t_2 \geq d - y, \quad t_2 \geq 0 \\ & && -b_i \leq (x_i)_1 \leq 0, \quad \text{for } i = 1, \dots, m \\ & && u_i + v_i \leq 1 \quad \text{for } i = 1, \dots, m \\ & && (-\beta_i(x_i)_1 + (3(x_i)_1 + (x_i)_2)/\alpha - \log(2), 1, u_i) \in K_{\text{exp}} \quad \text{for } i = 1, \dots, m \\ & && ((3(x_i)_1 + (x_i)_2)/\alpha - \log(2), 1, v_i) \in K_{\text{exp}} \quad \text{for } i = 1, \dots, m. \end{aligned}$$

Financial network routing problem: edges

- ▶ Most DEXs are implemented as *constant function market makers* (CFMMs)
- ▶ CFMMs are defined by their trading function $\varphi : \mathbb{R}_+^n \rightarrow \mathbb{R}$
- ▶ Maps reserves $R \in \mathbb{R}_+^n$ to a real number
- ▶ Is concave and increasing
- ▶ Accepts trade $\Delta \rightarrow \Lambda$ if $\varphi(R + \gamma\Delta - \Lambda) \geq \varphi(R)$.

Financial network routing problem: conic form

$$\begin{aligned} \text{maximize} \quad & c^T y - (1/2) \sum_{i=1}^n (p_1)_i - (1/2) \sum_{i=1}^m (t_1)_i \\ \text{subject to} \quad & (0.5, (p_1)_i, (p_2)_i) \in K_{\text{rot2}}, \quad i = 1, \dots, n \\ & p_1 \geq 0 \\ & p_2 \geq 0, \quad p_2 \geq -y \\ & (0.5, (t_1)_i, (t_2)_i) \in K_{\text{rot2}}, \quad i = 1, \dots, m \\ & t_1 \geq 0 \\ & t_2 \geq 0, \quad (t_2)_i \geq -(\Lambda_i - \Delta_i) \\ & (R + \gamma\Delta - \Lambda, \varphi(R)) \in K_{\text{pow}}(w_i), \quad i = 1, \dots, m_1 \\ & (-3\varphi(R), R + \gamma\Delta - \Lambda) \in K_{\text{geomean}}, \quad i = m_1 + 1, \dots, m \\ & \Delta_i, \Lambda_i \geq 0, \quad i = 1, \dots, m, \end{aligned}$$

Appendix

A bit of history

More theory

- Flow calculus

- Nonconvex flows

- Theory Miscellanea

Motivating example details

Numerical experiment details

Additional algorithmic considerations

Works cited

Full dual problem

- ▶ $\bar{U}$ and $\bar{V}_i$ introduce implicit nonnegativity constraints
- ▶ Dual problem with these explicit is:

$$\begin{aligned} &\text{minimize} && \bar{U}(\nu) + \sum_{i=1}^m \left(\bar{V}_i(\eta_i - A_i^T \nu) + f_i(\eta_i) \right) \\ &\text{subject to} && \nu \geq 0, \quad \eta_i \geq A_i^T \nu, \quad i = 1, \dots, m. \end{aligned}$$

- ▶ Letting $\mu = (\nu, \eta)$, a change of variables gives

$$\begin{aligned} &\text{minimize} && g(F^{-1} \tilde{\mu}) \\ &\text{subject to} && \tilde{\mu} \geq 0, \end{aligned}$$

Two-node subproblems

- ▶ The arbitrage problem for two nodes is

$$f(\eta) = -\eta_1 w + \eta_2 h(w)$$

- ▶ This has optimality conditions

$$\eta_2 h^+(w^*) \leq \eta_1 \leq \eta_2 h^-(w^*)$$

- ▶ When differentiable, forward and reverse derivatives equal

Second-stage problem

- ▶ Assume U strictly concave (so y^* unique)
- ▶ Let S be the set of strictly concave allowable flows
- ▶ Second-stage problem:

$$\begin{aligned} & \text{minimize} && \|y^* - \sum_{i=1}^m A_i x_i\| \\ & \text{subject to} && x_i = \tilde{x}_i^*, && i \in S \\ & && x_i \in T_i \cup \partial f_i(\eta_i^*), && i \notin S. \end{aligned}$$

Appendix

A bit of history

More theory

- Flow calculus

- Nonconvex flows

- Theory Miscellanea

Motivating example details

Numerical experiment details

Additional algorithmic considerations

Works cited

Works cited I

-  Agrawal, Akshay, Stephen Boyd, Deepak Narayanan, Fiodar Kazhamiaka, and Matei Zaharia (2022). “Allocation of fungible resources via a fast, scalable price discovery method”. In: *Mathematical Programming Computation* 14.3, pp. 593–622.
-  Angeris, Guillermo, Tarun Chitra, Theo Diamandis, Alex Evans, and Kshitij Kulkarni (2023). “The geometry of constant function market makers”. In: *arXiv preprint arXiv:2308.08066*.
-  Bertsekas, Dimitri (1998). *Network optimization: continuous and discrete models*. Vol. 8. Athena Scientific.
-  Bertsekas, Dimitri and Robert Gallager (1992). *Data networks*. Athena Scientific.
-  Chiang, Mung, Steven H Low, A Robert Calderbank, and John C Doyle (2007). “Layering as optimization decomposition: A mathematical theory of network architectures”. In: *Proceedings of the IEEE* 95.1, pp. 255–312.
-  Diamandis, Theo and Guillermo Angeris (2024a). “Solving the Convex Flow Problem”. In: URL: theodiamandis.com/pdfs/papers/routing-algorithm.pdf.
-  Diamandis, Theo, Guillermo Angeris, and Alan Edelman (2024b). “Conic and Nonconvex Network Flows”. In: *in preparation*.

Works cited II

-  Diamandis, Theo, Max Resnick, Tarun Chitra, and Guillermo Angeris (2023). “An Efficient Algorithm for Optimal Routing Through Constant Function Market Makers”. In: *arXiv preprint arXiv:2302.04938*.
-  Kraning, Matt, Eric Chu, Javad Lavaei, Stephen Boyd, et al. (2013). “Dynamic network energy management via proximal message passing”. In: *Foundations and Trends® in Optimization* 1.2, pp. 73–126.
-  Nagurney, Anna and Deniz Besik (2022). “Spatial price equilibrium networks with flow-dependent arc multipliers”. In: *Optimization Letters* 16.8, pp. 2483–2500.
-  Roughgarden, Tim (2007). “Routing games”. In: *Algorithmic game theory* 18, pp. 459–484.
-  Schütz, Peter, Asgeir Tomasgard, and Shabbir Ahmed (2009). “Supply chain design under uncertainty using sample average approximation and dual decomposition”. In: *European journal of operational research* 199.2, pp. 409–419.
-  Stursberg, Paul Melvin (2019). “On the mathematics of energy system optimization”. PhD thesis. Technische Universität München.

Works cited III



Végh, László A (2014). “Concave generalized flows with applications to market equilibria”. In: *Mathematics of Operations Research* 39.2, pp. 573–596.



Xiao, Lin, Mikael Johansson, and Stephen P Boyd (2004). “Simultaneous routing and resource allocation via dual decomposition”. In: *IEEE Transactions on Communications* 52.7, pp. 1136–1144.