Online Decision-Making on Graphs

Smoothness and Side Observations

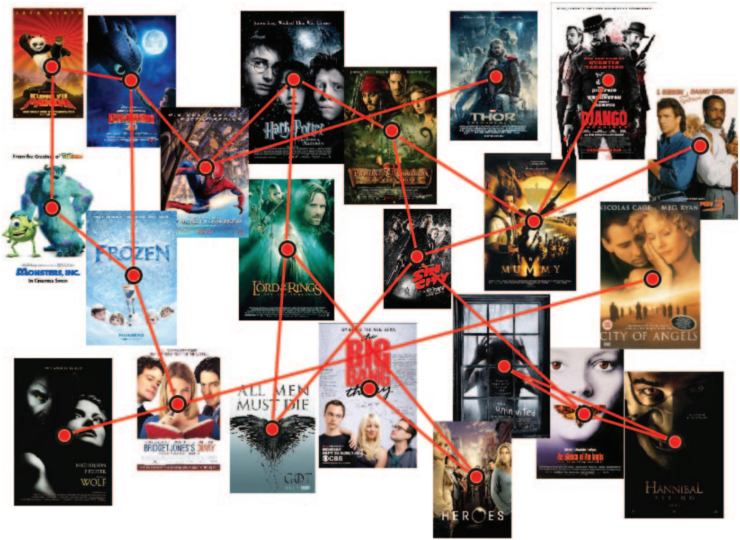
Michal Valko (SequeL INRIA)

joint work with
Shipra Agrawal (MSR India)
Tomáš Kocák (SequeL INRIA)
Branislav Kveton (Technicolor → Adobe)
Rémi Munos (SequeL INRIA/MSR → Google Deepmind)
Gergely Neu (SequeL INRIA)

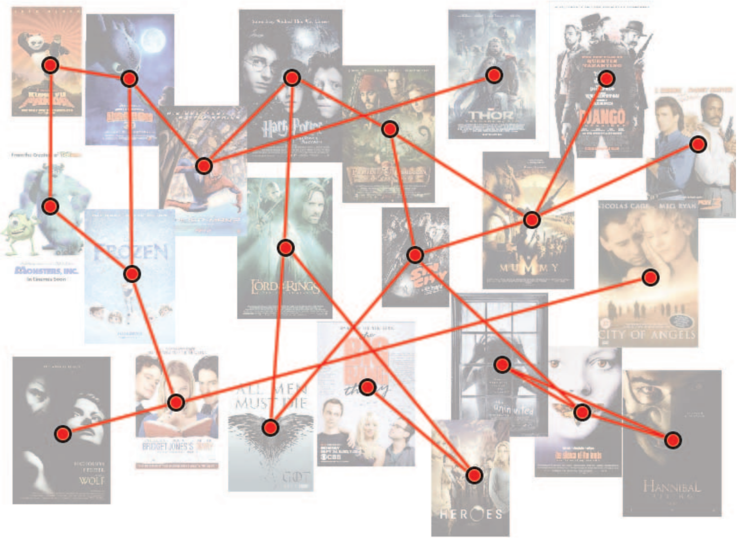
DaSciM, LIX, April 2015

Online Decision Making on Graphs

Online Decision Making on Graphs



Online Decision Making on Graphs



Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting
  - goal: maximize the sum of the outcomes

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting
  - goal: maximize the sum of the outcomes
  - application: recommender systems

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting
  - goal: maximize the sum of the outcomes
  - application: recommender systems

- Exploiting **smoothness**

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting
  - goal: maximize the sum of the outcomes
  - application: recommender systems
- Exploiting **smoothness**
  - fixed graph

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting
  - goal: maximize the sum of the outcomes
  - application: recommender systems
- Exploiting **smoothness**
  - fixed graph
  - iid outcomes

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting
  - goal: maximize the sum of the outcomes
  - application: recommender systems
- Exploiting **smoothness**
  - fixed graph
  - iid outcomes
  - neighboring nodes have similar outcomes

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting
  - goal: maximize the sum of the outcomes
  - application: recommender systems

- Exploiting **smoothness**
  - fixed graph
  - iid outcomes
  - neighboring nodes have similar outcomes

- Exploiting **side observations**

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting
  - goal: maximize the sum of the outcomes
  - application: recommender systems
- Exploiting **smoothness**
  - fixed graph
  - iid outcomes
  - neighboring nodes have similar outcomes
- Exploiting **side observations**
  - changing graph

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting
  - goal: maximize the sum of the outcomes
  - application: recommender systems

- Exploiting **smoothness**
  - fixed graph
  - iid outcomes
  - neighboring nodes have similar outcomes

- Exploiting **side observations**
  - changing graph
  - non-stochastic outcomes

Online Decision Making on Graphs: Smoothness

- Sequential decision making in structured settings
  - we are asked to pick a node (or a few nodes) in a graph
  - the graph encodes some **structural property** of the setting
  - goal: maximize the sum of the outcomes
  - application: recommender systems

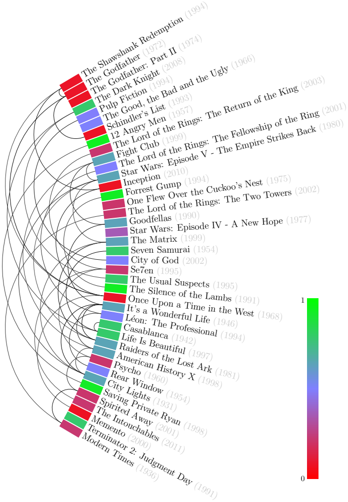
- Exploiting **smoothness**
  - fixed graph
  - iid outcomes
  - neighboring nodes have similar outcomes

- Exploiting **side observations**
  - changing graph
  - non-stochastic outcomes
  - side observations

Online Decision Making on Graphs

Movie recommendation: (in each time step)

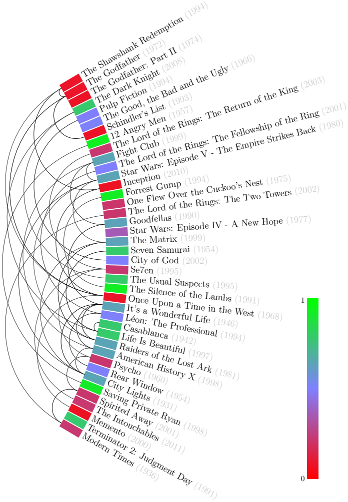
- Recommend movies to a **single user**.



Online Decision Making on Graphs

Movie recommendation: (in each time step)

- Recommend movies to a **single user**.
- Good prediction after a few steps ($T \ll N$).



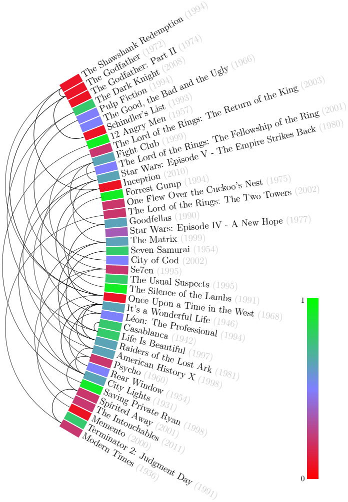
Online Decision Making on Graphs

Movie recommendation: (in each time step)

- Recommend movies to a **single user**.
- Good prediction after a few steps ($T \ll N$).

Goal:

- Maximize overall reward (sum of ratings).



Online Decision Making on Graphs

Movie recommendation: (in each time step)

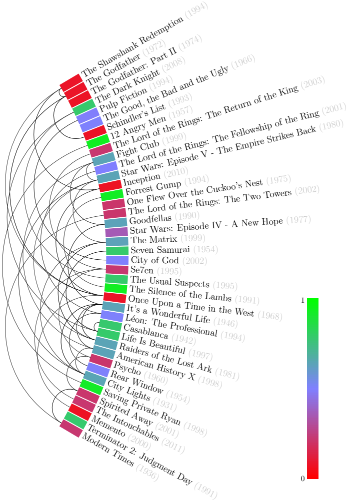
- Recommend movies to a **single user**.
- Good prediction after a few steps ($T \ll N$).

Goal:

- Maximize overall reward (sum of ratings).

Assumptions:

- Unknown reward function $f : V(G) \to \mathbb{R}$.
- Function f is **smooth** on a graph.
- Neighboring movies $\Rightarrow$ similar preferences.
- Similar preferences $\not\Rightarrow$ neighboring movies.



Recap: Smooth graph functions

- $\mathbf{f} = (f_1, \ldots, f_N)^\mathsf{T}$: Vector of function values.

Recap: Smooth graph functions

- $\mathbf{f} = (f_1, \dots, f_N)^\mathsf{T}$: Vector of function values.
- Let $\mathcal{L} = \mathbf{Q\Lambda Q}^\mathsf{T}$ be the eigendecomposition of the Laplacian.

Recap: Smooth graph functions

- $\mathbf{f} = (f_1, \ldots, f_N)^\mathsf{T}$: Vector of function values.
- Let $\mathcal{L} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\mathsf{T}$ be the eigendecomposition of the Laplacian.
  - Diagonal matrix $\boldsymbol{\Lambda}$ whose diagonal entries are eigenvalues of $\mathcal{L}$.

Recap: Smooth graph functions

- $\mathbf{f} = (f_1, \ldots, f_N)^\mathsf{T}$: Vector of function values.
- Let $\mathcal{L} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathsf{T}$ be the eigendecomposition of the Laplacian.
  - Diagonal matrix $\mathbf{\Lambda}$ whose diagonal entries are eigenvalues of $\mathcal{L}$.
  - Columns of $\mathbf{Q}$ are eigenvectors of $\mathcal{L}$.

Recap: Smooth graph functions

- $\mathbf{f} = (f_1, \dots, f_N)^\mathsf{T}$: Vector of function values.
- Let $\mathcal{L} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\mathsf{T}$ be the eigendecomposition of the Laplacian.
  - Diagonal matrix $\boldsymbol{\Lambda}$ whose diagonal entries are eigenvalues of $\mathcal{L}$.
  - Columns of $\mathbf{Q}$ are eigenvectors of $\mathcal{L}$.
  - Columns of $\mathbf{Q}$ form a basis.

Recap: Smooth graph functions

- $\mathbf{f} = (f_1, \ldots, f_N)^\intercal$: Vector of function values.
- Let $\mathcal{L} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\intercal$ be the eigendecomposition of the Laplacian.
  - Diagonal matrix $\boldsymbol{\Lambda}$ whose diagonal entries are eigenvalues of $\mathcal{L}$.
  - Columns of $\mathbf{Q}$ are eigenvectors of $\mathcal{L}$.
  - Columns of $\mathbf{Q}$ form a basis.
- $\boldsymbol{\alpha}^*$: Unique vector such that $\mathbf{Q}\boldsymbol{\alpha}^* = \mathbf{f}$ Note: $\mathbf{Q}^\intercal\mathbf{f} = \boldsymbol{\alpha}^*$

Recap: Smooth graph functions

- $\mathbf{f} = (f_1, \dots, f_N)^\mathsf{T}$: Vector of function values.
- Let $\mathcal{L} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathsf{T}$ be the eigendecomposition of the Laplacian.
  - Diagonal matrix $\mathbf{\Lambda}$ whose diagonal entries are eigenvalues of $\mathcal{L}$.
  - Columns of $\mathbf{Q}$ are eigenvectors of $\mathcal{L}$.
  - Columns of $\mathbf{Q}$ form a basis.
- $\boldsymbol{\alpha}^*$: Unique vector such that $\mathbf{Q}\boldsymbol{\alpha}^* = \mathbf{f}$ Note: $\mathbf{Q}^\mathsf{T}\mathbf{f} = \boldsymbol{\alpha}^*$

Recap: Smooth graph functions

- $\mathbf{f} = (f_1, \ldots, f_N)^\mathsf{T}$: Vector of function values.
- Let $\mathcal{L} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\mathsf{T}$ be the eigendecomposition of the Laplacian.
  - Diagonal matrix $\boldsymbol{\Lambda}$ whose diagonal entries are eigenvalues of $\mathcal{L}$.
  - Columns of $\mathbf{Q}$ are eigenvectors of $\mathcal{L}$.
  - Columns of $\mathbf{Q}$ form a basis.
- $\boldsymbol{\alpha}^*$: Unique vector such that $\mathbf{Q}\boldsymbol{\alpha}^* = \mathbf{f}$ Note: $\mathbf{Q}^\mathsf{T}\mathbf{f} = \boldsymbol{\alpha}^*$

$$S_G(\mathbf{f}) = \mathbf{f}^\mathsf{T}\mathcal{L}\mathbf{f} = \mathbf{f}^\mathsf{T}\mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\mathsf{T}\mathbf{f} = \boldsymbol{\alpha}^{*\mathsf{T}}\boldsymbol{\Lambda}\boldsymbol{\alpha}^* = \|\boldsymbol{\alpha}^*\|_{\boldsymbol{\Lambda}}^2 = \sum_{i=1}^{N} \lambda_i (\alpha_i^*)^2$$

Smoothness and regularization: Small value of

(a) $S_G(\mathbf{f})$ **(b)** $\boldsymbol{\Lambda}$ norm of $\boldsymbol{\alpha}^*$ **(c)** α_i^* for large λ_i

Online Learning Setting - Bandit Problem

Learning setting for a bandit algorithm π

- In each time t step choose a node $\pi(t)$.

Online Learning Setting - Bandit Problem

Learning setting for a bandit algorithm π

- In each time t step choose a node $\pi(t)$.
- the $\pi(t)$-th row $\mathbf{x}_{\pi(t)}$ of the matrix $\mathbf{Q}$ corresponds to the arm $\pi(t)$.

Online Learning Setting - Bandit Problem

Learning setting for a bandit algorithm π

- In each time t step choose a node $\pi(t)$.
- the $\pi(t)$-th row $\mathbf{x}_{\pi(t)}$ of the matrix $\mathbf{Q}$ corresponds to the arm $\pi(t)$.
- Obtain noisy reward $r_t = \mathbf{x}_{\pi(t)}^\mathsf{T}\boldsymbol{\alpha}^* + \varepsilon_t$. Note: $\mathbf{x}_{\pi(t)}^\mathsf{T}\boldsymbol{\alpha}^* = f_{\pi(t)}$
  - ε_t is R-sub-Gaussian noise. $\forall\xi \in \mathbb{R}, \mathbb{E}[e^{\xi\varepsilon_t}] \leq \exp\left(\xi^2 R^2/2\right)$

Online Learning Setting - Bandit Problem

Learning setting for a bandit algorithm π

- In each time t step choose a node $\pi(t)$.
- the $\pi(t)$-th row $\mathbf{x}_{\pi(t)}$ of the matrix $\mathbf{Q}$ corresponds to the arm $\pi(t)$.
- Obtain noisy reward $r_t = \mathbf{x}_{\pi(t)}^{\mathsf{T}}\boldsymbol{\alpha}^* + \varepsilon_t$. Note: $\mathbf{x}_{\pi(t)}^{\mathsf{T}}\boldsymbol{\alpha}^* = f_{\pi(t)}$
  - ε_t is R-sub-Gaussian noise. $\forall \xi \in \mathbb{R}, \mathbb{E}[e^{\xi\varepsilon_t}] \leq \exp\left(\xi^2 R^2/2\right)$
- Minimize cumulative regret

$$R_T = T \max_a \left(\mathbf{x}_a^{\mathsf{T}}\boldsymbol{\alpha}^*\right) - \sum_{t=1}^{T} \mathbf{x}_{\pi(t)}^{\mathsf{T}}\boldsymbol{\alpha}^*.$$

Online Learning Setting - Bandit Problem

Learning setting for a bandit algorithm π

- In each time t step choose a node $\pi(t)$.
- the $\pi(t)$-th row $\mathbf{x}_{\pi(t)}$ of the matrix $\mathbf{Q}$ corresponds to the arm $\pi(t)$.
- Obtain noisy reward $r_t = \mathbf{x}_{\pi(t)}^\mathsf{T}\boldsymbol{\alpha}^* + \varepsilon_t$. Note: $\mathbf{x}_{\pi(t)}^\mathsf{T}\boldsymbol{\alpha}^* = f_{\pi(t)}$
  - ε_t is R-sub-Gaussian noise. $\forall \xi \in \mathbb{R},\ \mathbb{E}[e^{\xi\varepsilon_t}] \leq \exp\left(\xi^2 R^2/2\right)$
- Minimize cumulative regret

$$R_T = T\max_a\left(\mathbf{x}_a^\mathsf{T}\boldsymbol{\alpha}^*\right) - \sum_{t=1}^{T}\mathbf{x}_{\pi(t)}^\mathsf{T}\boldsymbol{\alpha}^*.$$

Can't we just use *linear bandits*?

Online Decision Making on Graphs: Smoothness

- **Linear bandit algorithms**
  - **LinUCB** (Li et al., 2010)
    - Regret bound $\approx D\sqrt{T \ln T}$
  - **LinearTS** (Agrawal and Goyal, 2013)
    - Regret bound $\approx D\sqrt{T \ln N}$

Note: D is ambient dimension, in our case N, length of x_i.
Number of actions, e.g., all possible movies $\rightarrow$ **HUGE!**

Online Decision Making on Graphs: Smoothness

- **Linear bandit algorithms**
  - **LinUCB** (Li et al., 2010)
    - Regret bound $\approx D\sqrt{T \ln T}$
  - **LinearTS** (Agrawal and Goyal, 2013)
    - Regret bound $\approx D\sqrt{T \ln N}$

Note: D is ambient dimension, in our case N, length of x_i.
Number of actions, e.g., all possible movies $\rightarrow$ **HUGE!**

- **Spectral bandit algorithms**
  - **SpectralUCB** (Valko et al., ICML 2014)
    - Regret bound $\approx d\sqrt{T \ln T}$
  - **SpectralTS** (Kocák et al., AAAI 2014)
    - Regret bound $\approx d\sqrt{T \ln N}$

Note: d is effective dimension, usually much smaller than D.

Online Decision Making on Graphs: Smoothness

- **Linear bandit algorithms**
  - **LinUCB** (Li et al., 2010)
    - Regret bound $\approx D\sqrt{T \ln T}$
  - **LinearTS** (Agrawal and Goyal, 2013)
    - Regret bound $\approx D\sqrt{T \ln N}$

Note: D is ambient dimension, in our case N, length of x_i.
Number of actions, e.g., all possible movies $\rightarrow$ **HUGE!**

- **Spectral bandit algorithms**
  - **SpectralUCB** (Valko et al., ICML 2014)
    - Regret bound $\approx d\sqrt{T \ln T}$
    - Operations per step: $D^2 N$
  - **SpectralTS** (Kocák et al., AAAI 2014)
    - Regret bound $\approx d\sqrt{T \ln N}$
    - Operations per step: $D^2 + DN$

Note: d is effective dimension, usually much smaller than D.

Effective dimension

- **Effective dimension:** Largest d such that

$$(d-1)\lambda_d \leq \frac{T}{\log(1+T/\lambda)}.$$

Effective dimension

- **Effective dimension:** Largest d such that

$$(d-1)\lambda_d \leq \frac{T}{\log(1 + T/\lambda)}.$$

- Function of time horizon and graph properties

Effective dimension

- **Effective dimension:** Largest d such that

$$(d-1)\lambda_d \leq \frac{T}{\log(1+T/\lambda)}.$$

- Function of time horizon and graph properties
- λ_i : i-th smallest eigenvalue of $\mathbf{\Lambda}$.

Effective dimension

- **Effective dimension:** Largest d such that

$$(d-1)\lambda_d \leq \frac{T}{\log(1+T/\lambda)}.$$

- Function of time horizon and graph properties
- λ_i : i-th smallest eigenvalue of $\mathbf{\Lambda}$.
- λ: Regularization parameter of the algorithm.

Effective dimension

- **Effective dimension:** Largest d such that

$$(d-1)\lambda_d \leq \frac{T}{\log(1+T/\lambda)}.$$

- Function of time horizon and graph properties
- λ_i : i-th smallest eigenvalue of $\mathbf{\Lambda}$.
- λ: Regularization parameter of the algorithm.

Effective dimension

- **Effective dimension:** Largest d such that

$$(d-1)\lambda_d \leq \frac{T}{\log(1+T/\lambda)}.$$

- Function of time horizon and graph properties
- λ_i : i-th smallest eigenvalue of $\mathbf{\Lambda}$.
- λ: Regularization parameter of the algorithm.

Properties:

- d is small when the coefficients λ_i grow rapidly above time.

Effective dimension

- **Effective dimension:** Largest d such that

$$(d-1)\lambda_d \leq \frac{T}{\log(1+T/\lambda)}.$$

- Function of time horizon and graph properties
- λ_i : i-th smallest eigenvalue of $\mathbf{\Lambda}$.
- λ: Regularization parameter of the algorithm.

Properties:

- d is small when the coefficients λ_i grow rapidly above time.
- d is related to the number of "non-negligible" dimensions.

Effective dimension

- **Effective dimension:** Largest d such that

$$(d-1)\lambda_d \leq \frac{T}{\log(1+T/\lambda)}.$$

- Function of time horizon and graph properties
- λ_i : i-th smallest eigenvalue of $\mathbf{\Lambda}$.
- λ: Regularization parameter of the algorithm.

Properties:

- d is small when the coefficients λ_i grow rapidly above time.
- d is related to the number of "non-negligible" dimensions.
- Usually d is much smaller than D in real world graphs.

Effective dimension

- **Effective dimension:** Largest d such that

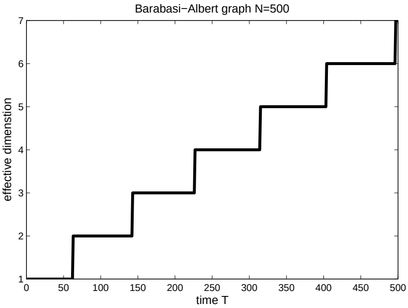
$$(d-1)\lambda_d \leq \frac{T}{\log(1+T/\lambda)}.$$

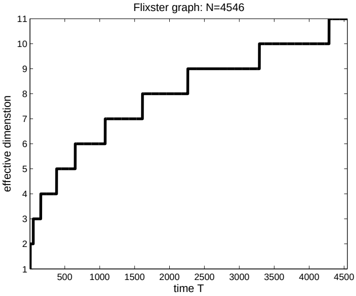
- Function of time horizon and graph properties
- λ_i : i-th smallest eigenvalue of $\mathbf{\Lambda}$.
- λ: Regularization parameter of the algorithm.

Properties:

- d is small when the coefficients λ_i grow rapidly above time.
- d is related to the number of "non-negligible" dimensions.
- Usually d is much smaller than D in real world graphs.
- Can be computed beforehand.

Effective dimension vs. Ambient dimension

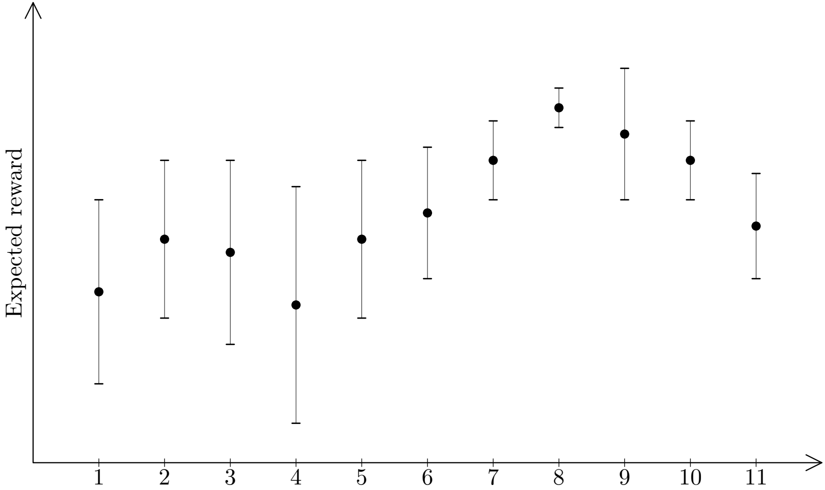




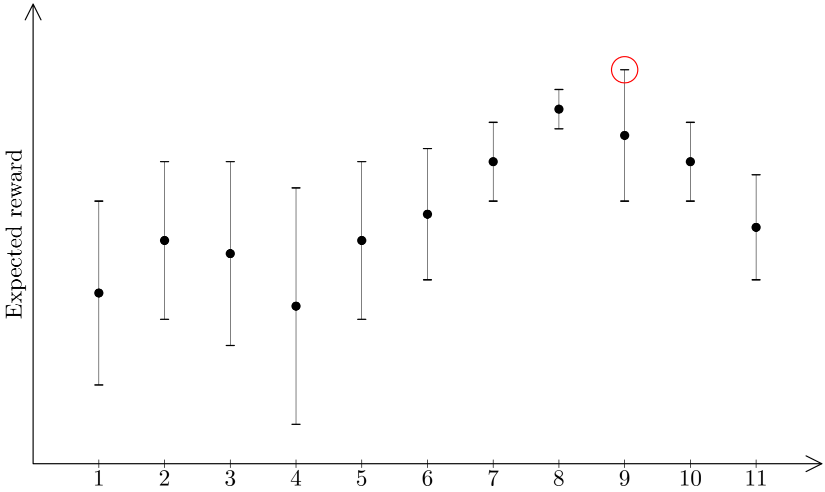
$$d \ll D$$

Note: In our setting $T < N = D$.

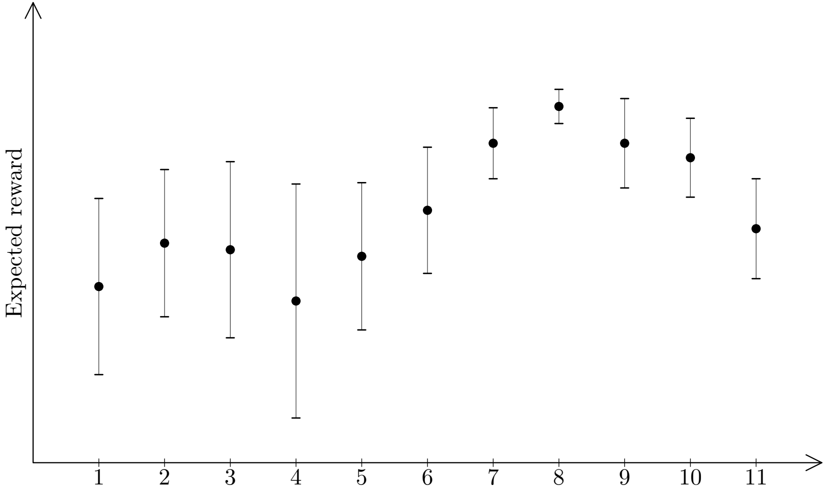
UCB style algorithms: Estimate



UCB style algorithms: Sample



UCB style algorithms: Estimate ...



SpectralUCB

Input:
N, T, $\{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\}$, λ, δ, R, C $\mathcal{L}$
Run:
$\mathbf{\Lambda} \leftarrow \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}$
$d \leftarrow \max\{d : (d-1)\lambda_d \leq T / \ln(1 + T/\lambda)\}$
for $t = 1$ **to** T **do**
Update the basis coefficients $\hat{\boldsymbol{\alpha}}$:
$\mathbf{X}_t \leftarrow [\mathbf{x}_{\pi(1)}, \ldots, \mathbf{x}_{\pi(t-1)}]^{\mathsf{T}}$
$\mathbf{r} \leftarrow [r_1, \ldots, r_{t-1}]^{\mathsf{T}}$
$\mathbf{V}_t \leftarrow \mathbf{X}_t \mathbf{X}_t^{\mathsf{T}} + \mathbf{\Lambda}$
$\hat{\boldsymbol{\alpha}}_t \leftarrow \mathbf{V}_t^{-1} \mathbf{X}_t^{\mathsf{T}} \mathbf{r}$
$c_t \leftarrow 2R\sqrt{d \ln(1 + t/\lambda) + 2\ln(1/\delta)} + C$
$\pi(t) \leftarrow \arg\max_a \left(\mathbf{x}_a^{\mathsf{T}} \hat{\boldsymbol{\alpha}} + c_t \|\mathbf{x}_a\|_{\mathbf{V}_t^{-1}} \right)$
Observe the reward r_t
end for

SpectralUCB

Input:
  N, T, $\{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\}$, λ, δ, R, C $\mathcal{L}$
Run:
  $\mathbf{\Lambda} \leftarrow \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}$
  $d \leftarrow \max\{d : (d-1)\lambda_d \leq T/\ln(1+T/\lambda)\}$
for $t = 1$ **to** T **do**
  Update the basis coefficients $\hat{\alpha}$:
  $\mathbf{X}_t \leftarrow [\mathbf{x}_{\pi(1)}, \ldots, \mathbf{x}_{\pi(t-1)}]^{\mathsf{T}}$
  $\mathbf{r} \leftarrow [r_1, \ldots, r_{t-1}]^{\mathsf{T}}$
  $\mathbf{V}_t \leftarrow \mathbf{X}_t \mathbf{X}_t^{\mathsf{T}} + \mathbf{\Lambda}$
  $\hat{\alpha}_t \leftarrow \mathbf{V}_t^{-1} \mathbf{X}_t^{\mathsf{T}} \mathbf{r}$
  $c_t \leftarrow 2R\sqrt{d \ln(1+t/\lambda) + 2\ln(1/\delta)} + C$
  $\pi(t) \leftarrow \arg\max_a \left(\mathbf{x}_a^{\mathsf{T}} \hat{\alpha} + c_t \|\mathbf{x}_a\|_{\mathbf{V}_t^{-1}}\right)$
  Observe the reward r_t
end for

SpectralUCB

Input:

N, T, $\{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\}$, λ, δ, R, C $\mathcal{L}$

Run:

$\mathbf{\Lambda} \leftarrow \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}$

$d \leftarrow \max\{d : (d-1)\lambda_d \leq T / \ln(1 + T/\lambda)\}$

for $t = 1$ **to** T **do**

Update the basis coefficients $\hat{\boldsymbol{\alpha}}$:

$\mathbf{X}_t \leftarrow [\mathbf{x}_{\pi(1)}, \ldots, \mathbf{x}_{\pi(t-1)}]^{\mathsf{T}}$

$\mathbf{r} \leftarrow [r_1, \ldots, r_{t-1}]^{\mathsf{T}}$

$\mathbf{V}_t \leftarrow \mathbf{X}_t \mathbf{X}_t^{\mathsf{T}} + \mathbf{\Lambda}$

$\hat{\boldsymbol{\alpha}}_t \leftarrow \mathbf{V}_t^{-1} \mathbf{X}_t^{\mathsf{T}} \mathbf{r}$

$c_t \leftarrow 2R\sqrt{d \ln(1 + t/\lambda) + 2\ln(1/\delta)} + C$

$\pi(t) \leftarrow \arg\max_a \left(\mathbf{x}_a^{\mathsf{T}} \hat{\boldsymbol{\alpha}} + c_t \|\mathbf{x}_a\|_{\mathbf{V}_t^{-1}}\right)$

Observe the reward r_t

end for

SpectralUCB

Input:

N, T, $\{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\}$, λ, δ, R, C $\mathcal{L}$

Run:

$\mathbf{\Lambda} \leftarrow \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}$

$d \leftarrow \max\{d : (d-1)\lambda_d \leq T/\ln(1+T/\lambda)\}$

for $t = 1$ **to** T **do**

Update the basis coefficients $\hat{\boldsymbol{\alpha}}$:

$\mathbf{X}_t \leftarrow [\mathbf{x}_{\pi(1)}, \ldots, \mathbf{x}_{\pi(t-1)}]^{\mathsf{T}}$

$\mathbf{r} \leftarrow [r_1, \ldots, r_{t-1}]^{\mathsf{T}}$

$\mathbf{V}_t \leftarrow \mathbf{X}_t \mathbf{X}_t^{\mathsf{T}} + \mathbf{\Lambda}$

$\hat{\boldsymbol{\alpha}}_t \leftarrow \mathbf{V}_t^{-1} \mathbf{X}_t^{\mathsf{T}} \mathbf{r}$

$c_t \leftarrow 2R\sqrt{d\ln(1+t/\lambda) + 2\ln(1/\delta)} + C$

$\pi(t) \leftarrow \arg\max_a \left(\mathbf{x}_a^{\mathsf{T}} \hat{\boldsymbol{\alpha}} + c_t \|\mathbf{x}_a\|_{\mathbf{V}_t^{-1}} \right)$

Observe the reward r_t

end for

SpectralUCB

Input:
N, T, $\{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\}$, λ, δ, R, C $\mathcal{L}$
Run:
$\mathbf{\Lambda} \leftarrow \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}$
$d \leftarrow \max\{d : (d-1)\lambda_d \leq T/\ln(1+T/\lambda)\}$
for $t = 1$ **to** T **do**
Update the basis coefficients $\hat{\boldsymbol{\alpha}}$:
$\mathbf{X}_t \leftarrow [\mathbf{x}_{\pi(1)}, \ldots, \mathbf{x}_{\pi(t-1)}]^{\mathsf{T}}$
$\mathbf{r} \leftarrow [r_1, \ldots, r_{t-1}]^{\mathsf{T}}$
$\mathbf{V}_t \leftarrow \mathbf{X}_t \mathbf{X}_t^{\mathsf{T}} + \mathbf{\Lambda}$
$\hat{\boldsymbol{\alpha}}_t \leftarrow \mathbf{V}_t^{-1} \mathbf{X}_t^{\mathsf{T}} \mathbf{r}$
$c_t \leftarrow 2R\sqrt{d \ln(1 + t/\lambda) + 2\ln(1/\delta)} + C$
$\pi(t) \leftarrow \arg\max_a \left(\mathbf{x}_a^{\mathsf{T}} \hat{\boldsymbol{\alpha}} + c_t \|\mathbf{x}_a\|_{\mathbf{V}_t^{-1}}\right)$
Observe the reward r_t
end for

SpectralUCB

Input:

N, T, $\{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\}$, λ, δ, R, C $\mathcal{L}$

Run:

$\mathbf{\Lambda} \leftarrow \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}$

$d \leftarrow \max\{d : (d-1)\lambda_d \leq T / \ln(1 + T/\lambda)\}$

for $t = 1$ **to** T **do**

Update the basis coefficients $\hat{\boldsymbol{\alpha}}$:

$\mathbf{X}_t \leftarrow [\mathbf{x}_{\pi(1)}, \ldots, \mathbf{x}_{\pi(t-1)}]^{\mathsf{T}}$

$\mathbf{r} \leftarrow [r_1, \ldots, r_{t-1}]^{\mathsf{T}}$

$\mathbf{V}_t \leftarrow \mathbf{X}_t \mathbf{X}_t^{\mathsf{T}} + \mathbf{\Lambda}$

$\hat{\boldsymbol{\alpha}}_t \leftarrow \mathbf{V}_t^{-1} \mathbf{X}_t^{\mathsf{T}} \mathbf{r}$

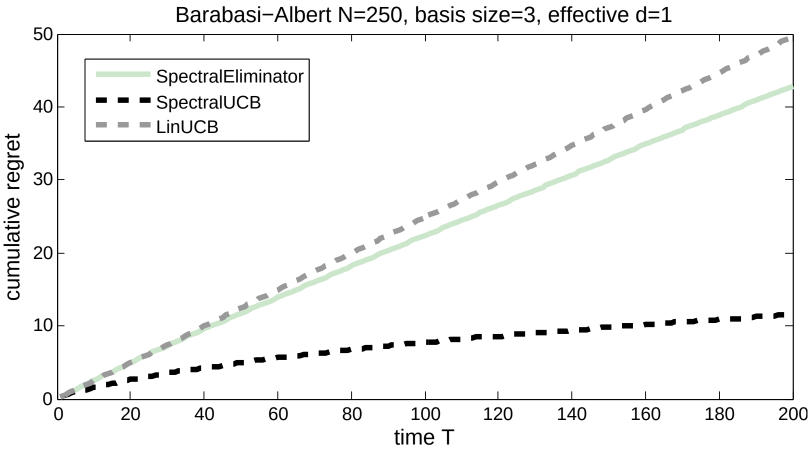
$c_t \leftarrow 2R\sqrt{d \ln(1 + t/\lambda) + 2\ln(1/\delta)} + C$

$\pi(t) \leftarrow \arg\max_a \left(\mathbf{x}_a^{\mathsf{T}} \hat{\boldsymbol{\alpha}} + c_t \|\mathbf{x}_a\|_{\mathbf{V}_t^{-1}} \right)$

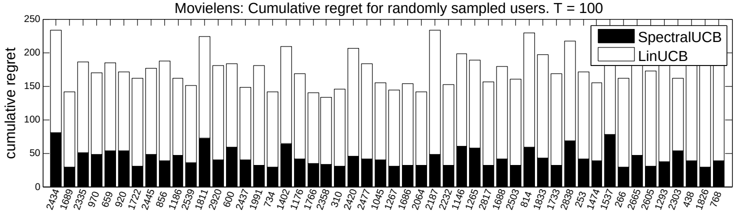
Observe the reward r_t

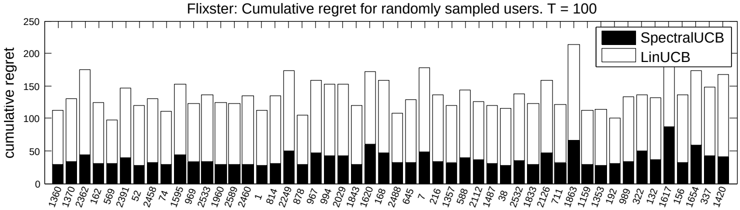
end for

SpectralUCB: Synthetic experiment



SpectralUCB: Real world experiment





SpectralUCB: Regret Bound

- d: Effective dimension.
- λ: Minimal eigenvalue of $\mathbf{\Lambda} = \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}$.
- C: Smoothness upper bound, $\|\boldsymbol{\alpha}^*\|_{\mathbf{\Lambda}} \leq C$.
- $\mathbf{x}_i^\mathsf{T} \boldsymbol{\alpha}^* \in [-1, 1]$ for all i.

The **cumulative regret** R_T of **SpectralUCB** is with probability $1 - \delta$ bounded as

$$R_T \leq \left(8R\sqrt{d \ln \frac{\lambda + T}{\lambda} + 2 \ln \frac{1}{\delta}} + 4C + 4\right) \sqrt{dT \ln \frac{\lambda + T}{\lambda}}.$$

$$R_T \approx d\sqrt{T \ln T}$$

SpectralUCB: Regret Bound

- Derivation of the confidence ellipsoid for $\hat{\boldsymbol{\alpha}}$ with probability $1 - \delta$.
  - Using analysis of OFUL (Abbasi-Yadkori et al., 2011)

$$|x^{\mathsf{T}}(\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*)| \leq \|x\|_{\mathbf{V}_t^{-1}} \left(R\sqrt{2\ln\left(\frac{|V_t|^{1/2}}{\delta|\boldsymbol{\Lambda}|^{1/2}}\right)} + C \right)$$

- Regret in one time step: $r_t = \mathbf{x}_*^{\mathsf{T}}\boldsymbol{\alpha}^* - \mathbf{x}_{\pi(t)}^{\mathsf{T}}\boldsymbol{\alpha}^* \leq 2c_t\|\mathbf{x}_{\pi(t)}\|_{\mathbf{V}_t^{-1}}$
- Cumulative regret:

$$R_T = \sum_{t=1}^{T} r_t \leq \sqrt{T\sum_{t=1}^{T} r_t^2} \leq 2(\ c_T + 1)\sqrt{2T\ln\frac{|\mathbf{V}_T|}{|\boldsymbol{\Lambda}|}}$$

SpectralUCB: Regret Bound

- Derivation of the confidence ellipsoid for $\hat{\boldsymbol{\alpha}}$ with probability $1-\delta$.
  - Using analysis of OFUL (Abbasi-Yadkori et al., 2011)

$$|x^{\mathsf{T}}(\hat{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^*)| \le \|x\|_{\mathbf{V}_t^{-1}} \left(R\sqrt{2\ln\left(\frac{|V_t|^{1/2}}{\delta|\boldsymbol{\Lambda}|^{1/2}}\right)}+C\right)$$

- Regret in one time step: $r_t = \mathbf{x}_*^{\mathsf{T}}\boldsymbol{\alpha}^* - \mathbf{x}_{\pi(t)}^{\mathsf{T}}\boldsymbol{\alpha}^* \le 2c_t\|\mathbf{x}_{\pi(t)}\|_{\mathbf{V}_t^{-1}}$
- Cumulative regret:

$$R_T = \sum_{t=1}^{T} r_t \le \sqrt{T\sum_{t=1}^{T} r_t^2} \le 2(c_T+1)\sqrt{2T\ln\frac{|\mathbf{V}_T|}{|\boldsymbol{\Lambda}|}}$$

SpectralUCB: Regret Bound

- Derivation of the confidence ellipsoid for $\hat{\boldsymbol{\alpha}}$ with probability $1-\delta$.
  - Using analysis of OFUL (Abbasi-Yadkori et al., 2011)

$$|x^{\mathsf{T}}(\hat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*)| \leq \|x\|_{\mathbf{V}_t^{-1}} \left(R\sqrt{2\ln\left(\frac{|V_t|^{1/2}}{\delta|\boldsymbol{\Lambda}|^{1/2}}\right)} + C \right)$$

- Regret in one time step: $r_t = \mathbf{x}_*^{\mathsf{T}}\boldsymbol{\alpha}^* - \mathbf{x}_{\pi(t)}^{\mathsf{T}}\boldsymbol{\alpha}^* \leq 2c_t \|\mathbf{x}_{\pi(t)}\|_{\mathbf{V}_t^{-1}}$
- Cumulative regret:

$$R_T = \sum_{t=1}^{T} r_t \leq \sqrt{T\sum_{t=1}^{T} r_t^2} \leq 2(c_T + 1)\sqrt{2T\ln\frac{|\mathbf{V}_T|}{|\boldsymbol{\Lambda}|}}$$

- Upperbound for $\ln(|\mathbf{V}_t|/|\boldsymbol{\Lambda}|)$

$$\ln\frac{|\mathbf{V}_t|}{|\boldsymbol{\Lambda}|} \leq \ln\frac{|\mathbf{V}_T|}{|\boldsymbol{\Lambda}|} \leq 2d\ln\left(\frac{\lambda+T}{\lambda}\right)$$

SpectralUCB: Regret Bound

Sylvester's determinant theorem:

$$|\mathbf{A} + \mathbf{x}\mathbf{x}^\mathsf{T}| = |\mathbf{A}||\mathbf{I} + \mathbf{A}^{-1}\mathbf{x}\mathbf{x}^\mathsf{T}| = |\mathbf{A}|(1 + \mathbf{x}^\mathsf{T}\mathbf{A}^{-1}\mathbf{x})$$

Goal:

- Upperbound determinant $|\mathbf{A} + \mathbf{x}\mathbf{x}^\mathsf{T}|$ for $\|\mathbf{x}\|_2 \leq 1$
- Upperbound $\mathbf{x}^\mathsf{T}\mathbf{A}^{-1}\mathbf{x}$

SpectralUCB: Regret Bound

Sylvester's determinant theorem:

$$|\mathbf{A} + \mathbf{x}\mathbf{x}^\intercal| = |\mathbf{A}||\mathbf{I} + \mathbf{A}^{-1}\mathbf{x}\mathbf{x}^\intercal| = |\mathbf{A}|(1 + \mathbf{x}^\intercal\mathbf{A}^{-1}\mathbf{x})$$

Goal:

- Upperbound determinant $|\mathbf{A} + \mathbf{x}\mathbf{x}^\intercal|$ for $\|\mathbf{x}\|_2 \leq 1$
- Upperbound $\mathbf{x}^\intercal\mathbf{A}^{-1}\mathbf{x}$

$$\mathbf{x}^\intercal\mathbf{A}^{-1}\mathbf{x} = \mathbf{x}^\intercal\mathbf{Q}\mathbf{\Lambda}^{-1}\mathbf{Q}^\intercal\mathbf{x} = \mathbf{y}^\intercal\mathbf{\Lambda}^{-1}\mathbf{y} = \sum_{i=1}^{N}\lambda_i^{-1}y_i^2$$

SpectralUCB: Regret Bound

Sylvester's determinant theorem:

$$|\mathbf{A} + \mathbf{x}\mathbf{x}^\mathsf{T}| = |\mathbf{A}||\mathbf{I} + \mathbf{A}^{-1}\mathbf{x}\mathbf{x}^\mathsf{T}| = |\mathbf{A}|(1 + \mathbf{x}^\mathsf{T}\mathbf{A}^{-1}\mathbf{x})$$

Goal:

- Upperbound determinant $|\mathbf{A} + \mathbf{x}\mathbf{x}^\mathsf{T}|$ for $\|\mathbf{x}\|_2 \leq 1$
- Upperbound $\mathbf{x}^\mathsf{T}\mathbf{A}^{-1}\mathbf{x}$

$$\mathbf{x}^\mathsf{T}\mathbf{A}^{-1}\mathbf{x} = \mathbf{x}^\mathsf{T}\mathbf{Q}\boldsymbol{\Lambda}^{-1}\mathbf{Q}^\mathsf{T}\mathbf{x} = \mathbf{y}^\mathsf{T}\boldsymbol{\Lambda}^{-1}\mathbf{y} = \sum_{i=1}^{N} \lambda_i^{-1} y_i^2$$

- $\|\mathbf{y}\|_2 \leq 1$.
- $\mathbf{y}$ is a canonical vector.
- $\mathbf{x} = \mathbf{Q}\mathbf{y}$ is an eigenvector of $\mathbf{A}$.

SpectralUCB: Regret Bound

Corollary: Determinant $|\mathbf{V}_T|$ of $\mathbf{V}_T = \mathbf{\Lambda} + \sum_{t=1}^{T} \mathbf{x}_t \mathbf{x}_t^{\mathsf{T}}$ is maximized when all $\mathbf{x}_t$ are aligned with axes.

$$
\begin{aligned}
|\mathbf{V}_T| &\leq \max_{\sum t_i = T} \prod (\lambda_i + t_i) \\
\ln \frac{|\mathbf{V}_T|}{|\mathbf{\Lambda}|} &\leq \max_{\sum t_i = T} \sum \ln \left(1 + \frac{t_i}{\lambda_i}\right) \\
\ln \frac{|\mathbf{V}_T|}{|\mathbf{\Lambda}|} &\leq \sum_{i=1}^{d} \ln \left(1 + \frac{T}{\lambda}\right) + \sum_{i=d+1}^{N} \ln \left(1 + \frac{t_i}{\lambda_{d+1}}\right) \\
&\leq d \ln \left(1 + \frac{T}{\lambda}\right) + \frac{T}{\lambda_{d+1}} \\
&\leq 2d \ln \left(1 + \frac{T}{\lambda}\right)
\end{aligned}
$$

SpectralUCB: Improving the running time

SpectralUCB: Improving the running time

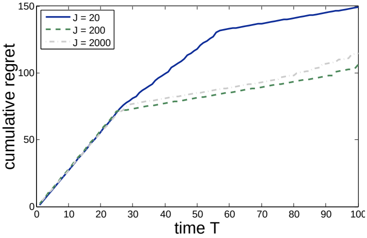
- **Reduced basis:** We only need first few eigenvectors.

SpectralUCB: Improving the running time

- **Reduced basis:** We only need first few eigenvectors.
- **Getting J eigenvectors:** $\mathcal{O}(Jm \log m)$ time for m edges

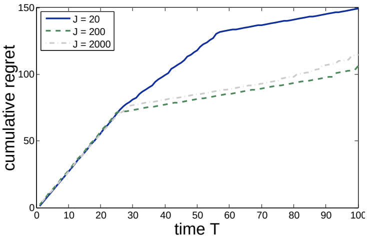
SpectralUCB: Improving the running time

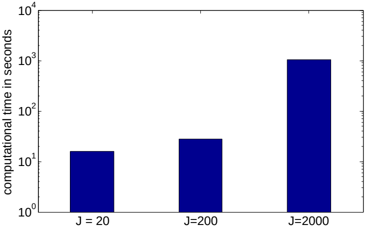
- **Reduced basis:** We only need first few eigenvectors.
- **Getting J eigenvectors:** $\mathcal{O}(Jm \log m)$ time for m edges
- Computationally less expensive, comparable performance.



SpectralUCB: Improving the running time

- **Reduced basis:** We only need first few eigenvectors.
- **Getting J eigenvectors:** $\mathcal{O}(Jm \log m)$ time for m edges
- Computationally less expensive, comparable performance.





SpectralUCB: How to make it even faster?

- UCB-style algorithms need to (re)-compute UCBs every t

SpectralUCB: How to make it even faster?

- UCB-style algorithms need to (re)-compute UCBs every t
- Can be a problem for large set of arms $\rightarrow D^2 N \rightarrow N^3$

SpectralUCB: How to make it even faster?

- UCB-style algorithms need to (re)-compute UCBs every t
- Can be a problem for large set of arms $\rightarrow D^2N \rightarrow N^3$
- Optimistic (UCB) approach vs. Thompson Sampling

SpectralUCB: How to make it even faster?

- UCB-style algorithms need to (re)-compute UCBs every t
- Can be a problem for large set of arms $\rightarrow D^2 N \rightarrow N^3$
- Optimistic (UCB) approach vs. Thompson Sampling
  - Play the arm maximizing probability of being the best

SpectralUCB: How to make it even faster?

- UCB-style algorithms need to (re)-compute UCBs every t
- Can be a problem for large set of arms $\rightarrow D^2 N \rightarrow N^3$
- Optimistic (UCB) approach vs. Thompson Sampling
  - Play the arm maximizing probability of being the best
    - Sample $\tilde{\boldsymbol{\mu}}$ from the distribution $\mathcal{N}(\hat{\boldsymbol{\mu}}, v^2\mathbf{B}^{-1})$

SpectralUCB: How to make it even faster?

- UCB-style algorithms need to (re)-compute UCBs every t
- Can be a problem for large set of arms $\rightarrow D^2N \rightarrow N^3$
- Optimistic (UCB) approach vs. Thompson Sampling
  - Play the arm maximizing probability of being the best
    - Sample $\tilde{\boldsymbol{\mu}}$ from the distribution $\mathcal{N}(\hat{\boldsymbol{\mu}}, v^2\mathbf{B}^{-1})$
    - Play arm which maximizes $\mathbf{b}^\top\tilde{\boldsymbol{\mu}}$ and observe reward

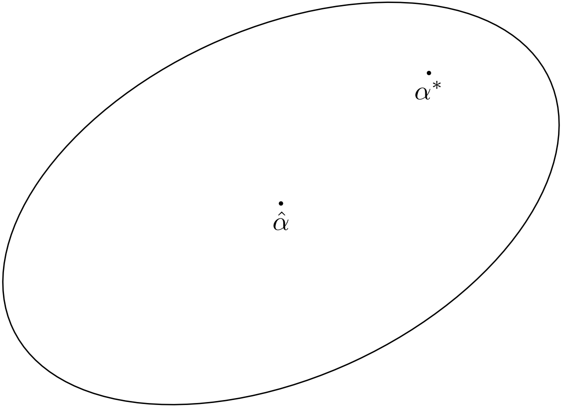
SpectralUCB: How to make it even faster?

- UCB-style algorithms need to (re)-compute UCBs every t
- Can be a problem for large set of arms $\rightarrow D^2N \rightarrow N^3$
- Optimistic (UCB) approach vs. Thompson Sampling
  - Play the arm maximizing probability of being the best
    - Sample $\tilde{\boldsymbol{\mu}}$ from the distribution $\mathcal{N}(\hat{\boldsymbol{\mu}}, v^2\mathbf{B}^{-1})$
    - Play arm which maximizes $\mathbf{b}^\mathsf{T}\tilde{\boldsymbol{\mu}}$ and observe reward
  - Compute posterior distribution according to reward received

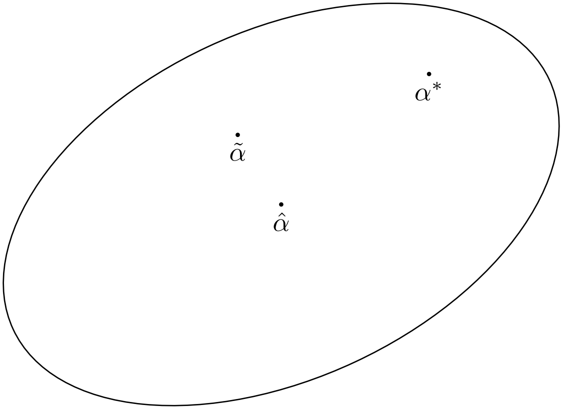
SpectralUCB: How to make it even faster?

- UCB-style algorithms need to (re)-compute UCBs every t
- Can be a problem for large set of arms $\to D^2 N \to N^3$
- Optimistic (UCB) approach vs. Thompson Sampling
  - Play the arm maximizing probability of being the best
    - Sample $\tilde{\boldsymbol{\mu}}$ from the distribution $\mathcal{N}(\hat{\boldsymbol{\mu}}, v^2\mathbf{B}^{-1})$
    - Play arm which maximizes $\mathbf{b}^\top\tilde{\boldsymbol{\mu}}$ and observe reward
  - Compute posterior distribution according to reward received
- Only requires $D^2 + DN \to N^2$ per step update

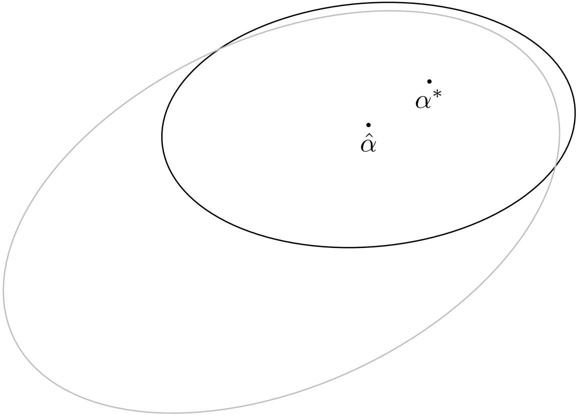
Thomson Sampling: Estimate



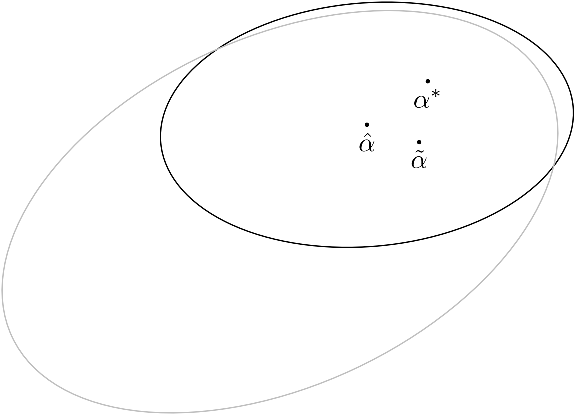
Thomson Sampling: Sample



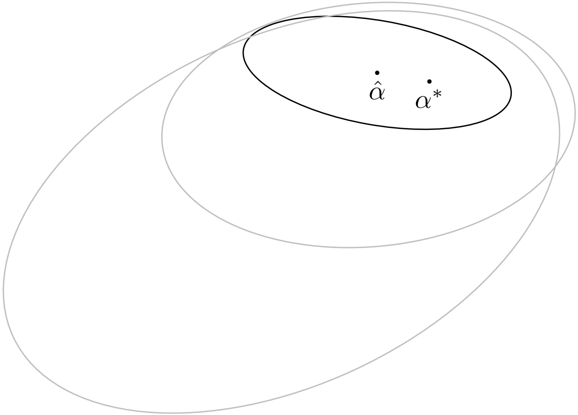
Thomson Sampling: Estimate



Thomson Sampling: Sample



Thomson Sampling: Estimate ...



SpectralTS for Graphs

Input:
N, T, $\{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\}$, λ, δ, R, C
Initialization:
$v = R\sqrt{6d\log((\lambda + T)/\delta\lambda)} + C$
$\hat{\boldsymbol{\alpha}} = 0_N$
$\mathbf{f} = 0_N$
$\mathbf{V} = \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}_N$
Run:
for $t = 1$ **to** T **do**
Sample $\tilde{\boldsymbol{\alpha}} \sim \mathcal{N}(\hat{\boldsymbol{\alpha}}, v^2\mathbf{V}^{-1})$
$\pi(t) \leftarrow \arg\max_a \mathbf{x}_a^\mathsf{T}\tilde{\boldsymbol{\alpha}}$
Observe a noisy reward $r(t) = \mathbf{x}_{\pi(t)}^\mathsf{T}\boldsymbol{\alpha}^* + \varepsilon_t$
$\mathbf{f} \leftarrow \mathbf{f} + \mathbf{x}_{\pi(t)} r(t)$
Update $\mathbf{V} \leftarrow \mathbf{V} + \mathbf{x}_{\pi(t)}\mathbf{x}_{\pi(t)}^\mathsf{T}$
Update $\hat{\boldsymbol{\alpha}} \leftarrow \mathbf{V}^{-1}\mathbf{f}$
end for

SpectralTS for Graphs

Input:

$N,\ T,\ \{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\},\ \lambda,\ \delta,\ R,\ C$

Initialization:

$v = R\sqrt{6d\log((\lambda + T)/\delta\lambda)} + C$

$\hat{\boldsymbol{\alpha}} = 0_N$

$\mathbf{f} = 0_N$

$\mathbf{V} = \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}_N$

Run:

for $t = 1$ **to** T **do**

Sample $\tilde{\boldsymbol{\alpha}} \sim \mathcal{N}(\hat{\boldsymbol{\alpha}}, v^2\mathbf{V}^{-1})$

$\pi(t) \leftarrow \arg\max_a \mathbf{x}_a^{\mathsf{T}}\tilde{\boldsymbol{\alpha}}$

Observe a noisy reward $r(t) = \mathbf{x}_{\pi(t)}^{\mathsf{T}}\boldsymbol{\alpha}^* + \varepsilon_t$

$\mathbf{f} \leftarrow \mathbf{f} + \mathbf{x}_{\pi(t)} r(t)$

Update $\mathbf{V} \leftarrow \mathbf{V} + \mathbf{x}_{\pi(t)}\mathbf{x}_{\pi(t)}^{\mathsf{T}}$

Update $\hat{\boldsymbol{\alpha}} \leftarrow \mathbf{V}^{-1}\mathbf{f}$

end for

SpectralTS for Graphs

Input:
  $N, T, \{\boldsymbol{\Lambda}_{\mathcal{L}}, \mathbf{Q}\}, \lambda, \delta, R, C$
Initialization:
  $v = R\sqrt{6d\log((\lambda + T)/\delta\lambda)} + C$
  $\hat{\boldsymbol{\alpha}} = 0_N$
  $\mathbf{f} = 0_N$
  $\mathbf{V} = \boldsymbol{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}_N$
Run:
for $t = 1$ **to** T **do**
  Sample $\tilde{\boldsymbol{\alpha}} \sim \mathcal{N}(\hat{\boldsymbol{\alpha}}, v^2\mathbf{V}^{-1})$
  $\pi(t) \leftarrow \arg\max_a \mathbf{x}_a^{\mathsf{T}}\tilde{\boldsymbol{\alpha}}$
  Observe a noisy reward $r(t) = \mathbf{x}_{\pi(t)}^{\mathsf{T}}\boldsymbol{\alpha}^* + \varepsilon_t$
  $\mathbf{f} \leftarrow \mathbf{f} + \mathbf{x}_{\pi(t)} r(t)$
  Update $\mathbf{V} \leftarrow \mathbf{V} + \mathbf{x}_{\pi(t)}\mathbf{x}_{\pi(t)}^{\mathsf{T}}$
  Update $\hat{\boldsymbol{\alpha}} \leftarrow \mathbf{V}^{-1}\mathbf{f}$
end for

SpectralTS for Graphs

Input:
  $N,\ T,\ \{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\},\ \lambda,\ \delta,\ R,\ C$
Initialization:
  $v = R\sqrt{6d\log((\lambda+T)/\delta\lambda)} + C$
  $\hat{\boldsymbol{\alpha}} = 0_N$
  $\mathbf{f} = 0_N$
  $\mathbf{V} = \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}_N$
Run:
for $t = 1$ **to** T **do**
  Sample $\tilde{\boldsymbol{\alpha}} \sim \mathcal{N}(\hat{\boldsymbol{\alpha}}, v^2\mathbf{V}^{-1})$
  $\pi(t) \leftarrow \arg\max_a \mathbf{x}_a^{\mathsf{T}}\tilde{\boldsymbol{\alpha}}$
  Observe a noisy reward $r(t) = \mathbf{x}_{\pi(t)}^{\mathsf{T}}\boldsymbol{\alpha}^* + \varepsilon_t$
  $\mathbf{f} \leftarrow \mathbf{f} + \mathbf{x}_{\pi(t)} r(t)$
  Update $\mathbf{V} \leftarrow \mathbf{V} + \mathbf{x}_{\pi(t)}\mathbf{x}_{\pi(t)}^{\mathsf{T}}$
  Update $\hat{\boldsymbol{\alpha}} \leftarrow \mathbf{V}^{-1}\mathbf{f}$
end for

SpectralTS for Graphs

Input:
  N, T, $\{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\}$, λ, δ, R, C
Initialization:
  $v = R\sqrt{6d\log((\lambda + T)/\delta\lambda) + C}$
  $\hat{\boldsymbol{\alpha}} = 0_N$
  $\mathbf{f} = 0_N$
  $\mathbf{V} = \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}_N$
Run:
for $t = 1$ **to** T **do**
  Sample $\tilde{\boldsymbol{\alpha}} \sim \mathcal{N}(\hat{\boldsymbol{\alpha}}, v^2\mathbf{V}^{-1})$
  $\pi(t) \leftarrow \arg\max_a \mathbf{x}_a^{\mathsf{T}}\tilde{\boldsymbol{\alpha}}$
  Observe a noisy reward $r(t) = \mathbf{x}_{\pi(t)}^{\mathsf{T}}\boldsymbol{\alpha}^* + \varepsilon_t$
  $\mathbf{f} \leftarrow \mathbf{f} + \mathbf{x}_{\pi(t)} r(t)$
  Update $\mathbf{V} \leftarrow \mathbf{V} + \mathbf{x}_{\pi(t)}\mathbf{x}_{\pi(t)}^{\mathsf{T}}$
  Update $\hat{\boldsymbol{\alpha}} \leftarrow \mathbf{V}^{-1}\mathbf{f}$
end for

SpectralTS: Regret bound

- d: Effective dimension.
- λ: Minimal eigenvalue of $\mathbf{\Lambda} = \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}$.
- C: Smoothness upper bound, $\|\boldsymbol{\alpha}^*\|_{\mathbf{\Lambda}} \leq C$.
- $\mathbf{x}_i^{\mathsf{T}} \boldsymbol{\alpha}^* \in [-1, 1]$ for all i.

The **cumulative regret** R_T of **SpectralTS** is with probability $1 - \delta$ bounded as

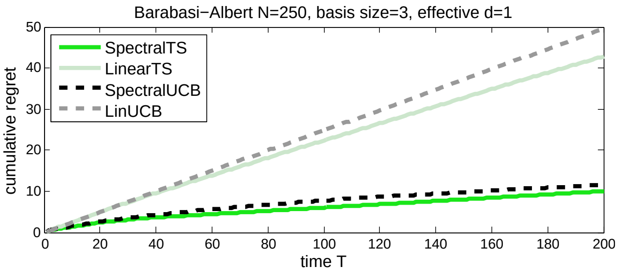
$$\mathcal{R}_T \leq \frac{11g}{p} \sqrt{\frac{4 + 4\lambda}{\lambda} dT \log \frac{\lambda + T}{\lambda}} + \frac{1}{T} + \frac{g}{p} \left(\frac{11}{\sqrt{\lambda}} + 2 \right) \sqrt{2T \log \frac{2}{\delta}},$$

where $p = 1/(4e\sqrt{\pi})$ and

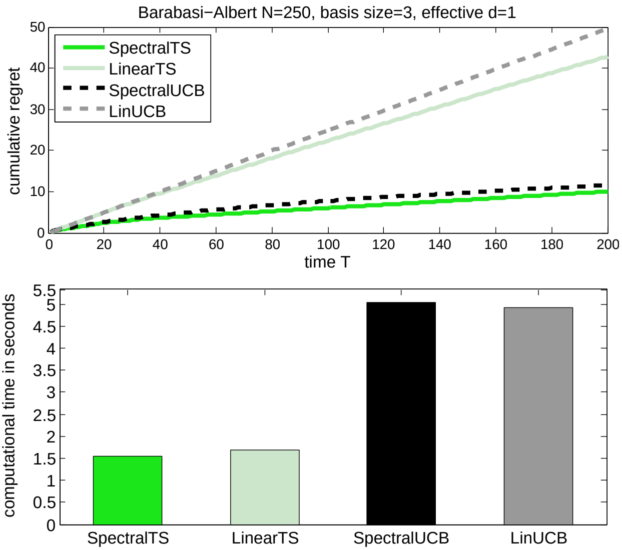
$$g = \sqrt{4 \log TN} \left(R \sqrt{6d \log \left(\frac{\lambda + T}{\delta \lambda} \right)} + C \right) + R \sqrt{2d \log \left(\frac{(\lambda + T)T^2}{\delta \lambda} \right)} + C.$$

$$R_T \approx d\sqrt{T \log N}$$

Spectral Bandits: Synthetic experiment

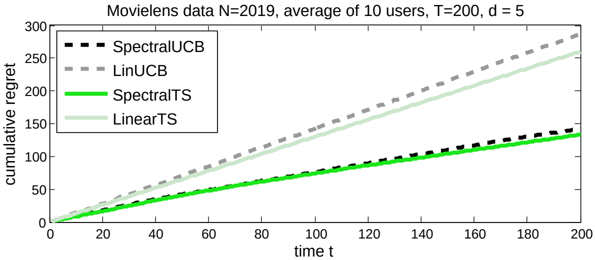


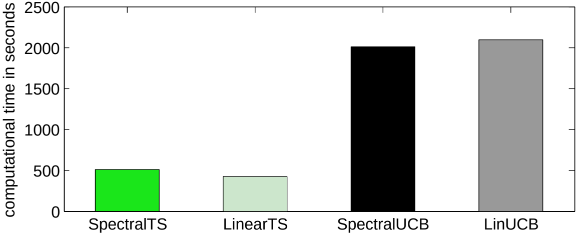
Spectral Bandits: Synthetic experiment



Spectral Bandits: Real world experiment

MovieLens dataset of 6k users who rated one million movies.





Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$
- **SpectralTS**

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$
- **SpectralTS**
  - Regret bound $\approx d\sqrt{T \ln N}$

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$
- **SpectralTS**
  - Regret bound $\approx d\sqrt{T \ln N}$
  - Computationally more efficient.

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$
- **SpectralTS**
  - Regret bound $\approx d\sqrt{T \ln N}$
  - Computationally more efficient.
- **SpectralEliminator**

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$
- **SpectralTS**
  - Regret bound $\approx d\sqrt{T \ln N}$
  - Computationally more efficient.
- **SpectralEliminator**
  - Regret bound $\approx \sqrt{dT \ln T}$

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$
- **SpectralTS**
  - Regret bound $\approx d\sqrt{T \ln N}$
  - Computationally more efficient.
- **SpectralEliminator**
  - Regret bound $\approx \sqrt{dT \ln T}$
  - Better upper, empirically does not seem to work well (yet)

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$
- **SpectralTS**
  - Regret bound $\approx d\sqrt{T \ln N}$
  - Computationally more efficient.
- **SpectralEliminator**
  - Regret bound $\approx \sqrt{dT \ln T}$
  - Better upper, empirically does not seem to work well (yet)

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$
- **SpectralTS**
  - Regret bound $\approx d\sqrt{T \ln N}$
  - Computationally more efficient.
- **SpectralEliminator**
  - Regret bound $\approx \sqrt{dT \ln T}$
  - Better upper, empirically does not seem to work well (yet)
- Bounds scale with **effective dimension** $d \ll D$.

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$
- **SpectralTS**
  - Regret bound $\approx d\sqrt{T \ln N}$
  - Computationally more efficient.
- **SpectralEliminator**
  - Regret bound $\approx \sqrt{dT \ln T}$
  - Better upper, empirically does not seem to work well (yet)
- Bounds scale with **effective dimension** $d \ll D$.

Spectral Bandits Summary

- Spectral bandit setting **(smooth graph functions)**.
- **SpectralUCB**
  - Regret bound $\approx d\sqrt{T \ln T}$
- **SpectralTS**
  - Regret bound $\approx d\sqrt{T \ln N}$
  - Computationally more efficient.
- **SpectralEliminator**
  - Regret bound $\approx \sqrt{dT \ln T}$
  - Better upper, empirically does not seem to work well (yet)
- Bounds scale with **effective dimension** $d \ll D$.

SpectralEliminator: Pseudocode

Input:

N : the number of nodes, T : the number of pulls

$\{\mathbf{\Lambda}_{\mathcal{L}}, \mathbf{Q}\}$ spectral basis of $\mathcal{L}$

λ : regularization parameter

$\beta, \{t_j\}_j^J$ parameters of the elimination and phases

$A_1 \leftarrow \{\mathbf{x}_1, \ldots, \mathbf{x}_K\}$.

for $j = 1$ **to** J **do**

$\mathbf{V}_{t_j} \leftarrow \gamma \mathbf{\Lambda}_{\mathcal{L}} + \lambda \mathbf{I}$

for $t = t_j$ **to** $\min(t_{j+1} - 1, T)$ **do**

Play $\mathbf{x}_t \in A_j$ with the largest width to observe r_t:

$\mathbf{x}_t \leftarrow \arg\max_{\mathbf{x} \in A_j} \|\mathbf{x}\|_{\mathbf{V}_t^{-1}}$

$\mathbf{V}_{t+1} \leftarrow \mathbf{V}_t + \mathbf{x}_t \mathbf{x}_t^\mathsf{T}$

end for

Eliminate the arms that are not promising:

$\hat{\alpha}_t \leftarrow \mathbf{V}_t^{-1}[\mathbf{x}_{t_j}, \ldots, \mathbf{x}_t][r_{t_j}, \ldots, r_t]^\mathsf{T}$

$A_{j+1} \leftarrow \left\{\mathbf{x} \in A_j, \langle \hat{\alpha}_t, \mathbf{x} \rangle + \|\mathbf{x}\|_{V_t^{-1}} \beta \geq \max_{\mathbf{x} \in A_j} \left[\langle \hat{\alpha}_t, \mathbf{x} \rangle - \|\mathbf{x}\|_{\mathbf{V}_t^{-1}} \beta\right]\right\}$

end for

Graph bandits: Side observations

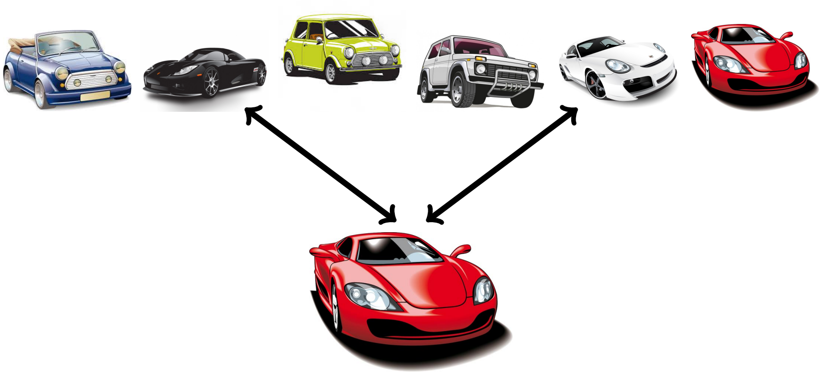
Example 1: undirected observations

Graph bandits: Side observations

Example 1: undirected observations

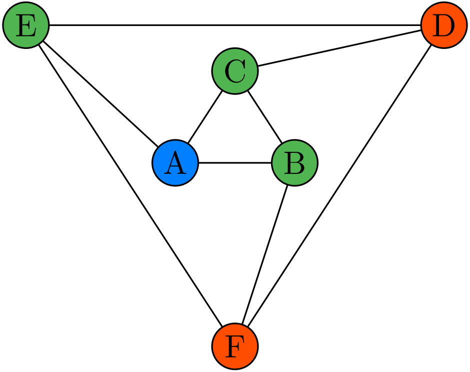
Graph bandits: Side observations

Example 1: undirected observations



Graph bandits: Side observations

Example 1: Graph Representation



Graph bandits: Side observations

Example 2: Directed observation

Graph bandits: Side observations

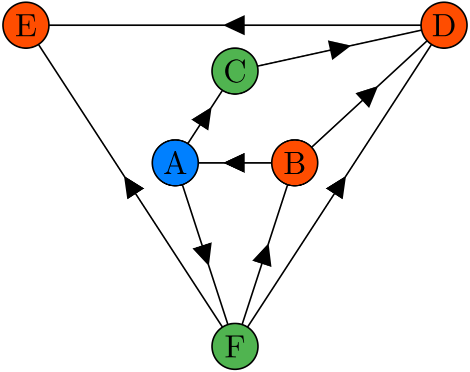
Example 2: Directed observation

Graph bandits: Side observations

Example 2: Directed observation

Graph bandits: Side observations

Example 2



Graph bandits: Side observations

Learning setting

In each time step $t = 1, \ldots, T$

- **Environment (adversary):**
  - Privately assigns losses to actions
  - Generates an observation graph

Graph bandits: Side observations

Learning setting

In each time step $t = 1, \ldots, T$

- **Environment (adversary):**
  - Privately assigns losses to actions
  - Generates an observation graph
    - Undirected / Directed

Graph bandits: Side observations

Learning setting

In each time step $t = 1, \ldots, T$

- **Environment (adversary):**
  - Privately assigns losses to actions
  - Generates an observation graph
    - Undirected / Directed
    - Disclosed / Not disclosed

Graph bandits: Side observations

Learning setting

In each time step $t = 1, \ldots, T$

- **Environment (adversary):**
  - Privately assigns losses to actions
  - Generates an observation graph
    - Undirected / Directed
    - Disclosed / Not disclosed
- **Learner:**
  - Plays action $I_t \in [N]$
  - Obtain loss ℓ_{t,I_t} of action played
  - Observe losses of neighbors of I_t

Graph bandits: Side observations

Learning setting

In each time step $t = 1, \ldots, T$

- **Environment (adversary):**
  - Privately assigns losses to actions
  - Generates an observation graph
    - Undirected / Directed
    - Disclosed / Not disclosed
- **Learner:**
  - Plays action $I_t \in [N]$
  - Obtain loss ℓ_{t,I_t} of action played
  - Observe losses of neighbors of I_t
    - Graph: disclosed

Graph bandits: Side observations

Learning setting

In each time step $t = 1, \ldots, T$

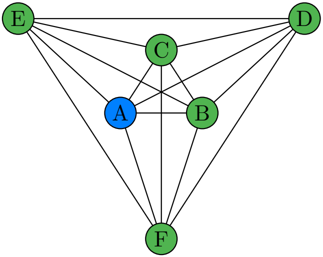
- **Environment (adversary):**
  - Privately assigns losses to actions
  - Generates an observation graph
    - Undirected / Directed
    - Disclosed / Not disclosed
- **Learner:**
  - Plays action $I_t \in [N]$
  - Obtain loss ℓ_{t,I_t} of action played
  - Observe losses of neighbors of I_t
    - Graph: disclosed
- **Performance measure:** Total expected regret

$$R_T = \max_{i\in[N]} \mathbb{E}\left[\sum_{t=1}^{T}(\ell_{t,I_t} - \ell_{t,i})\right]$$

Graph bandits: Typical settings

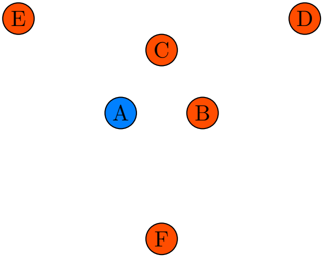
Full Information setting

- Pick an action (e.g. action A)
- Observe losses of all actions
- $R_T = \widetilde{\mathcal{O}}(\sqrt{T})$



Bandit setting

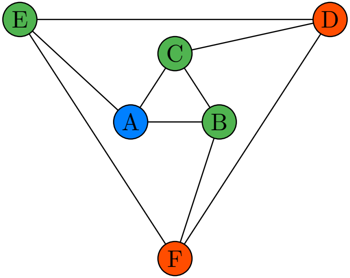
- Pick an action (e.g. action A)
- Observe loss of a chosen action
- $R_T = \widetilde{\mathcal{O}}(\sqrt{NT})$



Graph bandits: Side observation - Undirected case

Side observation (Undirected case)

- Pick an action (e.g. action A)
- Observe losses of neighbors



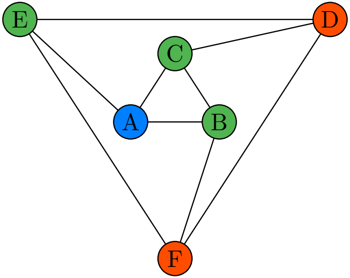
Graph bandits: Side observation - Undirected case

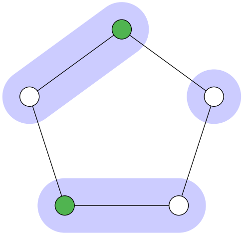
Side observation (Undirected case)

- Pick an action (e.g. action A)
- Observe losses of neighbors

Mannor and Shamir (ELP algorithm)

- Need to know graph
- Clique decomposition (c cliques)
- $R_T = \widetilde{\mathcal{O}}(\sqrt{cT})$





Graph bandits: Side observation - Undirected case

Side observation (Undirected case)

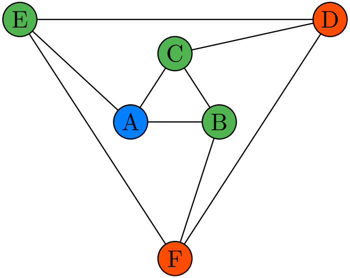
- Pick an action (e.g. action A)
- Observe losses of neighbors

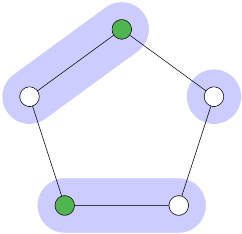
Mannor and Shamir (ELP algorithm)

- Need to know graph
- Clique decomposition (c cliques)
- $R_T = \widetilde{\mathcal{O}}(\sqrt{cT})$

Alon, Cesa-Bianchi, Gentile, Mansour

- No need to know graph
- Independence set of α actions
- $R_T = \widetilde{\mathcal{O}}(\sqrt{\alpha T})$

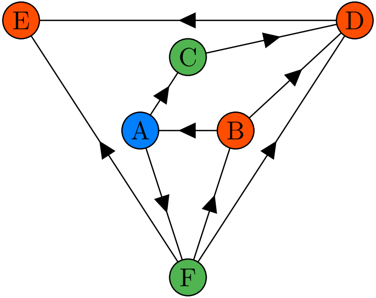




Graph bandits: Side observation - Directed case

Side observation **(Directed case)**

- Pick an action (e.g. action A)
- Observe losses of neighbors



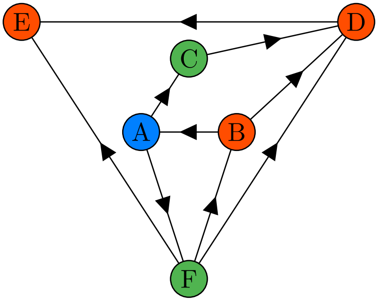
Graph bandits: Side observation - Directed case

Side observation (Directed case)

- Pick an action (e.g. action A)
- Observe losses of neighbors

Alon, Cesa-Bianchi, Gentile, Mansour

- **Exp3-DOM**
- Need to know graph
- Need to find dominating set
- $R_T = \widetilde{\mathcal{O}}(\sqrt{\alpha T})$



Graph bandits: Side observation - Directed case

Side observation (Directed case)

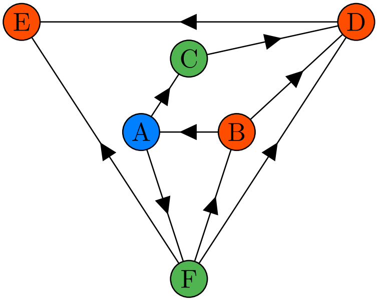
- Pick an action (e.g. action A)
- Observe losses of neighbors

Alon, Cesa-Bianchi, Gentile, Mansour

- **Exp3-DOM**
- Need to know graph
- Need to find dominating set
- $R_T = \widetilde{\mathcal{O}}(\sqrt{\alpha T})$

Exp3-IX - Kocák et. al

- No need to know graph
- $R_T = \widetilde{\mathcal{O}}(\sqrt{\alpha T})$



Reminder: Exp3 algorithms in general

- **Compute weights** using loss estimates $\hat{\ell}_{t,i}$.

$$w_{t,i} = \exp\left(-\eta \sum_{s=1}^{t-1} \hat{\ell}_{s,i}\right)$$

Reminder: Exp3 algorithms in general

- **Compute weights** using loss estimates $\hat{\ell}_{t,i}$.

$$w_{t,i} = \exp\left(-\eta \sum_{s=1}^{t-1} \hat{\ell}_{s,i}\right)$$

- **Play action** I_t such that

$$\mathbb{P}(I_t = i) = p_{t,i} = \frac{w_{t,i}}{W_t} = \frac{w_{t,i}}{\sum_{j=1}^{N} w_{t,j}}$$

Reminder: Exp3 algorithms in general

- **Compute weights** using loss estimates $\hat{\ell}_{t,i}$.

$$w_{t,i} = \exp\left(-\eta \sum_{s=1}^{t-1} \hat{\ell}_{s,i}\right)$$

- **Play action** I_t such that

$$\mathbb{P}(I_t = i) = p_{t,i} = \frac{w_{t,i}}{W_t} = \frac{w_{t,i}}{\sum_{j=1}^{N} w_{t,j}}$$

- **Update loss estimates** (using observability graph)

Reminder: Exp3 algorithms in general

- **Compute weights** using loss estimates $\hat{\ell}_{t,i}$.

$$w_{t,i} = \exp\left(-\eta \sum_{s=1}^{t-1} \hat{\ell}_{s,i}\right)$$

- **Play action** I_t such that

$$\mathbb{P}(I_t = i) = p_{t,i} = \frac{w_{t,i}}{W_t} = \frac{w_{t,i}}{\sum_{j=1}^{N} w_{t,j}}$$

- **Update loss estimates** (using observability graph)

How the algorithms approach to bias variance tradeoff?

Bias variance tradeoff approaches

- Approach of Mixing

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions
    - $\mathbf{p}_t' = (1-\gamma)\mathbf{p}_t + \gamma \mathbf{s}_t$ – mixed distribution

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions
    - $\mathbf{p}'_t = (1-\gamma)\mathbf{p}_t + \gamma\mathbf{s}_t$ – mixed distribution
    - $\mathbf{s}_t$ – probability distribution which supports exploration

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions
    - $\mathbf{p}'_t = (1-\gamma)\mathbf{p}_t + \gamma\mathbf{s}_t$ – mixed distribution
    - $\mathbf{s}_t$ – probability distribution which supports exploration
  - Loss estimates $\hat{\ell}_{t,i}$ are unbiased

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions
    - $\mathbf{p}'_t = (1-\gamma)\mathbf{p}_t + \gamma\mathbf{s}_t$ – mixed distribution
    - $\mathbf{s}_t$ – probability distribution which supports exploration
  - Loss estimates $\hat{\ell}_{t,i}$ are unbiased
- Approach of **I**mplicit e**X**ploration (**IX**)

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions
    - $\mathbf{p}'_t = (1-\gamma)\mathbf{p}_t + \gamma\mathbf{s}_t$ – mixed distribution
    - $\mathbf{s}_t$ – probability distribution which supports exploration
  - Loss estimates $\hat{\ell}_{t,i}$ are unbiased

- Approach of Implicit eXploration (**IX**)
  - Bias loss estimates $\hat{\ell}_{t,i}$

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions
    - $\mathbf{p}'_t = (1-\gamma)\mathbf{p}_t + \gamma\mathbf{s}_t$ – mixed distribution
    - $\mathbf{s}_t$ – probability distribution which supports exploration
  - Loss estimates $\hat{\ell}_{t,i}$ are unbiased

- Approach of Implicit eXploration (**IX**)
  - Bias loss estimates $\hat{\ell}_{t,i}$
    - Biased loss estimates $\Longrightarrow$ biased weights

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions
    - $\mathbf{p}'_t = (1-\gamma)\mathbf{p}_t + \gamma\mathbf{s}_t$ – mixed distribution
    - $\mathbf{s}_t$ – probability distribution which supports exploration
  - Loss estimates $\hat{\ell}_{t,i}$ are unbiased

- Approach of Implicit eXploration (**IX**)
  - Bias loss estimates $\hat{\ell}_{t,i}$
    - Biased loss estimates $\implies$ biased weights
    - Biased weights $\implies$ biased probability distribution

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions
    - $\mathbf{p}'_t = (1-\gamma)\mathbf{p}_t + \gamma\mathbf{s}_t$ – mixed distribution
    - $\mathbf{s}_t$ – probability distribution which supports exploration
  - Loss estimates $\hat{\ell}_{t,i}$ are unbiased

- Approach of Implicit eXploration (**IX**)
  - Bias loss estimates $\hat{\ell}_{t,i}$
    - Biased loss estimates $\implies$ biased weights
    - Biased weights $\implies$ biased probability distribution
  - No need for mixing

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions
    - $\mathbf{p}'_t = (1-\gamma)\mathbf{p}_t + \gamma\mathbf{s}_t$ – mixed distribution
    - $\mathbf{s}_t$ – probability distribution which supports exploration
  - Loss estimates $\hat{\ell}_{t,i}$ are unbiased
- Approach of Implicit eXploration (IX)
  - Bias loss estimates $\hat{\ell}_{t,i}$
    - Biased loss estimates $\implies$ biased weights
    - Biased weights $\implies$ biased probability distribution
  - No need for mixing

Bias variance tradeoff approaches

- Approach of Mixing
  - Bias sampling distribution $\mathbf{p}_t$ over actions
    - $\mathbf{p}'_t = (1-\gamma)\mathbf{p}_t + \gamma\mathbf{s}_t$ – mixed distribution
    - $\mathbf{s}_t$ – probability distribution which supports exploration
  - Loss estimates $\hat{\ell}_{t,i}$ are unbiased

- Approach of Implicit eXploration (**IX**)
  - Bias loss estimates $\hat{\ell}_{t,i}$
    - Biased loss estimates $\Longrightarrow$ biased weights
    - Biased weights $\Longrightarrow$ biased probability distribution
  - No need for mixing

Big difference in graph feedback case!

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$\mathbf{s}_t = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} \left(s_{t,j} + \sum_{k\in N_{t,j}} s_{t,k} \right) \right] = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} q_{t,j} \right]$$

- $q_{t,j}$ – probability that loss of j is observed according to $\mathbf{s}_t$

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$\mathbf{s}_t = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} \left(s_{t,j} + \sum_{k\in N_{t,j}} s_{t,k} \right) \right] = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} q_{t,j} \right]$$

- $q_{t,j}$ – probability that loss of j is observed according to $\mathbf{s}_t$

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$\mathbf{s}_t = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} \left(s_{t,j} + \sum_{k\in N_{t,j}} s_{t,k} \right) \right] = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} q_{t,j} \right]$$

- $q_{t,j}$ – probability that loss of j is observed according to $\mathbf{s}_t$
- **Computation of $\mathbf{s}_t$**

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$\mathbf{s}_t = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} \left(s_{t,j} + \sum_{k\in N_{t,j}} s_{t,k} \right) \right] = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} q_{t,j} \right]$$

- $q_{t,j}$ – probability that loss of j is observed according to $\mathbf{s}_t$
- **Computation of $\mathbf{s}_t$**
  - Graph needs to be disclosed

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$\mathbf{s}_t = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} \left(s_{t,j} + \sum_{k\in N_{t,j}} s_{t,k} \right) \right] = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} q_{t,j} \right]$$

- $q_{t,j}$ – probability that loss of j is observed according to $\mathbf{s}_t$
- **Computation of $\mathbf{s}_t$**
  - Graph needs to be disclosed
  - Solving simple linear program

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$\mathbf{s}_t = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} \left(s_{t,j} + \sum_{k\in N_{t,j}} s_{t,k} \right) \right] = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} q_{t,j} \right]$$

- $q_{t,j}$ – probability that loss of j is observed according to $\mathbf{s}_t$
- **Computation of $\mathbf{s}_t$**
  - Graph needs to be disclosed
  - Solving simple linear program
- Needs to know graph before playing an action

Graph bandits: Mannor and Shamir - ELP algorithm

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$\mathbf{s}_t = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} \left(s_{t,j} + \sum_{k\in N_{t,j}} s_{t,k} \right) \right] = \arg\max_{\mathbf{s}_t} \left[\min_{j\in[N]} q_{t,j} \right]$$

- $q_{t,j}$ – probability that loss of j is observed according to $\mathbf{s}_t$
- **Computation of $\mathbf{s}_t$**
  - Graph needs to be disclosed
  - Solving simple linear program
- Needs to know graph before playing an action
- Graphs can be only undirected

Graph bandits: Alon, Cesa-Bianchi, Gentile, Mansour - Exp3-DOM

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates

Graph bandits: Alon, Cesa-Bianchi, Gentile, Mansour - Exp3-DOM

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing

Graph bandits: Alon, Cesa-Bianchi, Gentile, Mansour - Exp3-DOM

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

Graph bandits: Alon, Cesa-Bianchi, Gentile, Mansour - Exp3-DOM

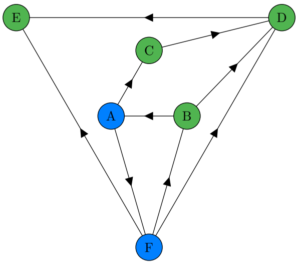
- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

Graph bandits: Alon, Cesa-Bianchi, Gentile, Mansour - Exp3-DOM

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$s_{t,i} = \begin{cases} \frac{1}{r} & \text{if } i \in R;\ |R| = r \\ 0 & \text{otherwise.} \end{cases}$$

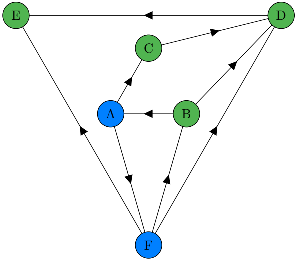
- R – dominating set of r elements



Graph bandits: Alon, Cesa-Bianchi, Gentile, Mansour - Exp3-DOM

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$s_{t,i} = \begin{cases} \frac{1}{r} & \text{if } i \in R;\ |R| = r \\ 0 & \text{otherwise.} \end{cases}$$

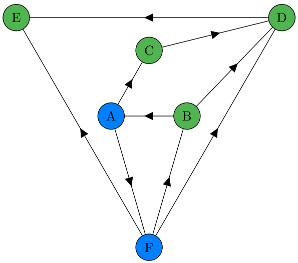


- R – dominating set of r elements
- $\mathbf{s}_t$ – uniform distribution over R

Graph bandits: Alon, Cesa-Bianchi, Gentile, Mansour - Exp3-DOM

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$s_{t,i} = \begin{cases} \frac{1}{r} & \text{if } i \in R;\ |R| = r \\ 0 & \text{otherwise.} \end{cases}$$

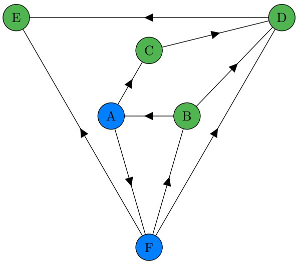


- R – dominating set of r elements
- $\mathbf{s}_t$ – uniform distribution over R
- Needs to know graph beforehand

Graph bandits: Alon, Cesa-Bianchi, Gentile, Mansour - Exp3-DOM

- $\mathbb{E}[\hat{\ell}_{t,i}] = \ell_{t,i}$ – unbiased loss estimates
- $p'_{t,i} = (1-\gamma)p_{t,i} + \gamma s_{t,i}$ – bias by mixing
- $\mathbf{s}_t = \{s_{t,1}, \ldots, s_{t,N}\}$ – probability distribution over the action set

$$s_{t,i} = \begin{cases} \frac{1}{r} & \text{if } i \in R;\ |R| = r \\ 0 & \text{otherwise.} \end{cases}$$



- R – dominating set of r elements
- $\mathbf{s}_t$ – uniform distribution over R
- Needs to know graph beforehand
- Graphs can be directed

Graph bandits: Comparison of loss estimates

Typical algorithms - loss estimates

$$\hat{\ell}_{t,i} = \begin{cases} \ell_{t,i}/o_{t,i} & \text{if } \ell_{t,i} \text{ is observed} \\ 0 & \text{otherwise.} \end{cases}$$

Graph bandits: Comparison of loss estimates

Typical algorithms - loss estimates

$$\hat{\ell}_{t,i} = \begin{cases} \ell_{t,i}/o_{t,i} & \text{if } \ell_{t,i} \text{ is observed} \\ 0 & \text{otherwise.} \end{cases}$$

$$\mathbb{E}[\hat{\ell}_{t,i}] = \frac{\ell_{t,i}}{o_{t,i}} o_{t,i} + 0(1 - o_{t,i}) = \ell_{t,i}$$

Graph bandits: Comparison of loss estimates

Typical algorithms - loss estimates

$$\hat{\ell}_{t,i} = \begin{cases} \ell_{t,i}/o_{t,i} & \text{if } \ell_{t,i} \text{ is observed} \\ 0 & \text{otherwise.} \end{cases}$$

$$\mathbb{E}[\hat{\ell}_{t,i}] = \frac{\ell_{t,i}}{o_{t,i}} o_{t,i} + 0(1 - o_{t,i}) = \ell_{t,i}$$

Exp3-IX - loss estimates

$$\hat{\ell}_{t,i} = \begin{cases} \ell_{t,i}/(o_{t,i} + \gamma) & \text{if } \ell_{t,i} \text{ is observed} \\ 0 & \text{otherwise.} \end{cases}$$

Graph bandits: Comparison of loss estimates

Typical algorithms - loss estimates

$$\hat{\ell}_{t,i} = \begin{cases} \ell_{t,i}/o_{t,i} & \text{if } \ell_{t,i} \text{ is observed} \\ 0 & \text{otherwise.} \end{cases}$$

$$\mathbb{E}[\hat{\ell}_{t,i}] = \frac{\ell_{t,i}}{o_{t,i}} o_{t,i} + 0(1 - o_{t,i}) = \ell_{t,i}$$

Exp3-IX - loss estimates

$$\hat{\ell}_{t,i} = \begin{cases} \ell_{t,i}/(o_{t,i} + \gamma) & \text{if } \ell_{t,i} \text{ is observed} \\ 0 & \text{otherwise.} \end{cases}$$

$$\mathbb{E}[\hat{\ell}_{t,i}] = \frac{\ell_{t,i}}{o_{t,i} + \gamma} o_{t,i} + 0(1 - o_{t,i}) = \ell_{t,i} - \ell_{t,i} \frac{\gamma}{o_{t,i} + \gamma} \leq \ell_{t,i}$$

Graph bandits: Comparison of loss estimates

Typical algorithms - loss estimates

$$\hat{\ell}_{t,i} = \begin{cases} \ell_{t,i}/o_{t,i} & \text{if } \ell_{t,i} \text{ is observed} \\ 0 & \text{otherwise.} \end{cases}$$

$$\mathbb{E}[\hat{\ell}_{t,i}] = \frac{\ell_{t,i}}{o_{t,i}} o_{t,i} + 0(1 - o_{t,i}) = \ell_{t,i}$$

Exp3-IX - loss estimates

$$\hat{\ell}_{t,i} = \begin{cases} \ell_{t,i}/(o_{t,i} + \gamma) & \text{if } \ell_{t,i} \text{ is observed} \\ 0 & \text{otherwise.} \end{cases}$$

$$\mathbb{E}[\hat{\ell}_{t,i}] = \frac{\ell_{t,i}}{o_{t,i} + \gamma} o_{t,i} + 0(1 - o_{t,i}) = \ell_{t,i} - \ell_{t,i} \frac{\gamma}{o_{t,i} + \gamma} \leq \ell_{t,i}$$

No mixing!

Analysis of Exp3 algorithms in general

- Evolution of W_{t+1}/W_t

Analysis of Exp3 algorithms in general

- Evolution of W_{t+1}/W_t

$$\frac{1}{\eta}\log\frac{W_{t+1}}{W_t} \le \frac{1}{\eta}\log\left(1-\eta\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i}+\frac{\eta^2}{2}\sum_{i=1}^{N} p_{t,i}(\hat{\ell}_{t,i})^2\right),$$

Analysis of Exp3 algorithms in general

- Evolution of W_{t+1}/W_t

$$\frac{1}{\eta}\log\frac{W_{t+1}}{W_t} \leq \frac{1}{\eta}\log\left(1-\eta\sum_{i=1}^{N}p_{t,i}\hat{\ell}_{t,i}+\frac{\eta^2}{2}\sum_{i=1}^{N}p_{t,i}(\hat{\ell}_{t,i})^2\right),$$

$$\sum_{i=1}^{N}p_{t,i}\hat{\ell}_{t,i} \leq \left[\frac{\log W_t}{\eta}-\frac{\log W_{t+1}}{\eta}\right]+\frac{\eta}{2}\sum_{i=1}^{N}p_{t,i}(\hat{\ell}_{t,i})^2$$

Analysis of Exp3 algorithms in general

- Evolution of W_{t+1}/W_t

$$\frac{1}{\eta}\log\frac{W_{t+1}}{W_t} \leq \frac{1}{\eta}\log\left(1-\eta\sum_{i=1}^{N}p_{t,i}\hat{\ell}_{t,i}+\frac{\eta^2}{2}\sum_{i=1}^{N}p_{t,i}(\hat{\ell}_{t,i})^2\right),$$

$$\sum_{i=1}^{N}p_{t,i}\hat{\ell}_{t,i} \leq \left[\frac{\log W_t}{\eta}-\frac{\log W_{t+1}}{\eta}\right]+\frac{\eta}{2}\sum_{i=1}^{N}p_{t,i}(\hat{\ell}_{t,i})^2$$

- Taking expectation and summing over time

Analysis of Exp3 algorithms in general

- Evolution of W_{t+1}/W_t

$$\frac{1}{\eta}\log\frac{W_{t+1}}{W_t} \leq \frac{1}{\eta}\log\left(1-\eta\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i} + \frac{\eta^2}{2}\sum_{i=1}^{N} p_{t,i}(\hat{\ell}_{t,i})^2\right),$$

$$\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i} \leq \left[\frac{\log W_t}{\eta} - \frac{\log W_{t+1}}{\eta}\right] + \frac{\eta}{2}\sum_{i=1}^{N} p_{t,i}(\hat{\ell}_{t,i})^2$$

- Taking expectation and summing over time

$$\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i}\right] - \mathbb{E}\left[\sum_{t=1}^{T}\hat{\ell}_{t,k}\right] \leq \mathbb{E}\left[\frac{\log N}{\eta}\right] + \mathbb{E}\left[\frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}(\hat{\ell}_{t,i})^2\right]$$

Graph bandits: Regret bound of Exp3-IX

$$\underbrace{\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i}\right]}_{A} - \underbrace{\mathbb{E}\left[\sum_{t=1}^{T}\hat{\ell}_{t,k}\right]}_{B} \leq \mathbb{E}\left[\frac{\log N}{\eta}\right] + \underbrace{\mathbb{E}\left[\frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\left(\hat{\ell}_{t,i}\right)^2\right]}_{C}$$

Graph bandits: Regret bound of Exp3-IX

$$\underbrace{\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i}\right]}_{A} - \underbrace{\mathbb{E}\left[\sum_{t=1}^{T}\hat{\ell}_{t,k}\right]}_{B} \leq \mathbb{E}\left[\frac{\log N}{\eta}\right] + \underbrace{\mathbb{E}\left[\frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}(\hat{\ell}_{t,i})^2\right]}_{C}$$

Lower bound of A (using definition of loss estimates)

$$\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i}\right] \geq \mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\ell_{t,i}\right] - \mathbb{E}\left[\gamma\sum_{t=1}^{T}\sum_{i=1}^{N} \frac{p_{t,i}}{o_{t,i}+\gamma}\right]$$

Graph bandits: Regret bound of Exp3-IX

$$\underbrace{\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i}\right]}_{A} - \underbrace{\mathbb{E}\left[\sum_{t=1}^{T}\hat{\ell}_{t,k}\right]}_{B} \leq \mathbb{E}\left[\frac{\log N}{\eta}\right] + \underbrace{\mathbb{E}\left[\frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}(\hat{\ell}_{t,i})^2\right]}_{C}$$

Lower bound of A (using definition of loss estimates)

$$\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i}\right] \geq \mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\ell_{t,i}\right] - \mathbb{E}\left[\gamma\sum_{t=1}^{T}\sum_{i=1}^{N}\frac{p_{t,i}}{o_{t,i}+\gamma}\right]$$

Lower bound of B (optimistic loss estimates: $\mathbb{E}[\hat{\ell}] < \mathbb{E}[\ell]$)

$$-\mathbb{E}\left[\sum_{t=1}^{T}\hat{\ell}_{t,k}\right] \geq -\mathbb{E}\left[\sum_{t=1}^{T}\ell_{t,k}\right]$$

Graph bandits: Regret bound of Exp3-IX

$$\underbrace{\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i}\right]}_{A} - \underbrace{\mathbb{E}\left[\sum_{t=1}^{T}\hat{\ell}_{t,k}\right]}_{B} \leq \mathbb{E}\left[\frac{\log N}{\eta}\right] + \underbrace{\mathbb{E}\left[\frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}(\hat{\ell}_{t,i})^2\right]}_{C}$$

Lower bound of A (using definition of loss estimates)

$$\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\hat{\ell}_{t,i}\right] \geq \mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}\ell_{t,i}\right] - \mathbb{E}\left[\gamma\sum_{t=1}^{T}\sum_{i=1}^{N} \frac{p_{t,i}}{o_{t,i}+\gamma}\right]$$

Lower bound of B (optimistic loss estimates: $\mathbb{E}[\hat{\ell}] < \mathbb{E}[\ell]$)

$$-\mathbb{E}\left[\sum_{t=1}^{T}\hat{\ell}_{t,k}\right] \geq -\mathbb{E}\left[\sum_{t=1}^{T}\ell_{t,k}\right]$$

Upper bound of C (using definition of loss estimates)

$$\mathbb{E}\left[\frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} p_{t,i}(\hat{\ell}_{t,i})^2\right] \leq \mathbb{E}\left[\frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} \frac{p_{t,i}}{o_{t,i}+\gamma}\right]$$

Graph bandits: Regret bound of Exp3-IX

Upper bound on regret Exp3-IX

$$R_T \leq \frac{\log N}{\eta} + \left(\frac{\eta}{2} + \gamma\right) \sum_{t=1}^{T} \mathbb{E}\left[\sum_{i=1}^{N} \frac{p_{t,i}}{o_{t,i} + \gamma}\right]$$

Graph bandits: Regret bound of Exp3-IX

Upper bound on regret Exp3-IX

$$R_T \leq \frac{\log N}{\eta} + \left(\frac{\eta}{2} + \gamma\right) \sum_{t=1}^{T} \mathbb{E}\left[\sum_{i=1}^{N} \frac{p_{t,i}}{o_{t,i} + \gamma}\right]$$

$$R_T \approx \mathcal{O}\left(\sqrt{\log N \sum_{t=1}^{T} \mathbb{E}\left[\sum_{i=1}^{N} \frac{p_{t,i}}{o_{t,i} + \gamma}\right]}\right)$$

Graph bandits: Regret bound of Exp3-IX

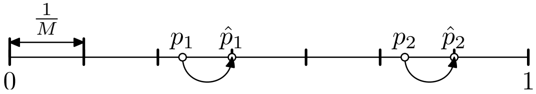
Graph lemma

- Graph G with $V(G) = \{1, \ldots, N\}$
- d_i^- – in-degree of vertex i
- α – independence set of G
- Turán's Theorem + induction

$$\sum_{i=1}^{N} \frac{1}{1 + d_i^-} \leq 2\alpha \log\left(1 + \frac{N}{\alpha}\right)$$

Graph bandits: Regret bound of Exp3-IX

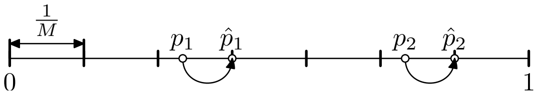
Discretization



$$\sum_{i=1}^{N} \frac{p_{t,i}}{o_{t,i} + \gamma} = \sum_{i=1}^{N} \frac{p_{t,i}}{p_{t,i} + \sum_{j \in N_i^-} p_{t,j} + \gamma} \leq \sum_{i=1}^{N} \frac{\hat{p}_{t,i}}{\hat{p}_{t,i} + \sum_{j \in N_i^-} \hat{p}_{t,j}} + 2$$

Graph bandits: Regret bound of Exp3-IX

Discretization

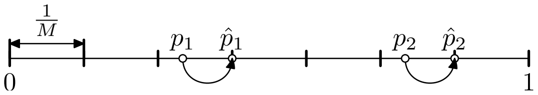


$$\sum_{i=1}^{N} \frac{p_{t,i}}{o_{t,i}+\gamma} = \sum_{i=1}^{N} \frac{p_{t,i}}{p_{t,i}+\sum_{j\in N_i^-} p_{t,j}+\gamma} \leq \sum_{i=1}^{N} \frac{\hat{p}_{t,i}}{\hat{p}_{t,i}+\sum_{j\in N_i^-} \hat{p}_{t,j}} + 2$$

Note: we set $M = \lceil N^2/\gamma \rceil$

Graph bandits: Regret bound of Exp3-IX

Discretization



$$\sum_{i=1}^{N} \frac{p_{t,i}}{o_{t,i} + \gamma} = \sum_{i=1}^{N} \frac{p_{t,i}}{p_{t,i} + \sum_{j \in N_i^-} p_{t,j} + \gamma} \leq \sum_{i=1}^{N} \frac{\hat{p}_{t,i}}{\hat{p}_{t,i} + \sum_{j \in N_i^-} \hat{p}_{t,j}} + 2$$

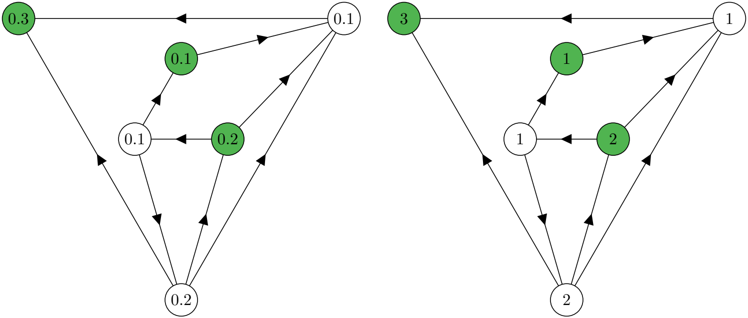
Note: we set $M = \lceil N^2/\gamma \rceil$

$$\sum_{i=1}^{N} \frac{\hat{p}_{t,i}}{\hat{p}_{t,i} + \sum_{j \in N_i^-} \hat{p}_{t,j}}$$

Graph bandits: Regret bound of Exp3-IX

$$\sum_{i=1}^{N} \frac{\hat{p}_{t,i}}{\hat{p}_{t,i} + \sum_{j \in N_i^-} \hat{p}_{t,j}}$$

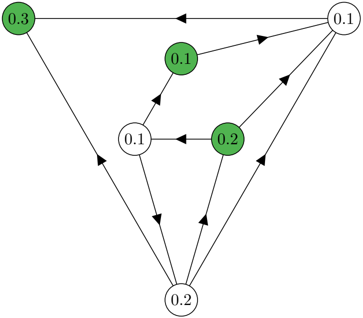
Example: let $M = 10$

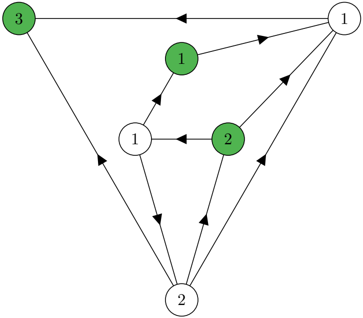


Graph bandits: Regret bound of Exp3-IX

$$\sum_{i=1}^{N} \frac{M\hat{p}_{t,i}}{M\hat{p}_{t,i} + \sum_{j \in N_i^-} M\hat{p}_{t,j}}$$

Example: let $M = 10$

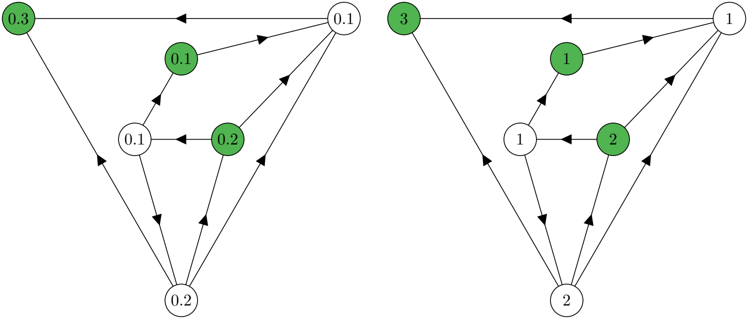




Graph bandits: Regret bound of Exp3-IX

$$\sum_{i=1}^{N} \frac{M\hat{p}_{t,i}}{M\hat{p}_{t,i} + \sum_{j \in N_i^-} M\hat{p}_{t,j}} = \sum_{i=1}^{N} \sum_{k \in C_i} \frac{1}{1 + d_k^-}$$

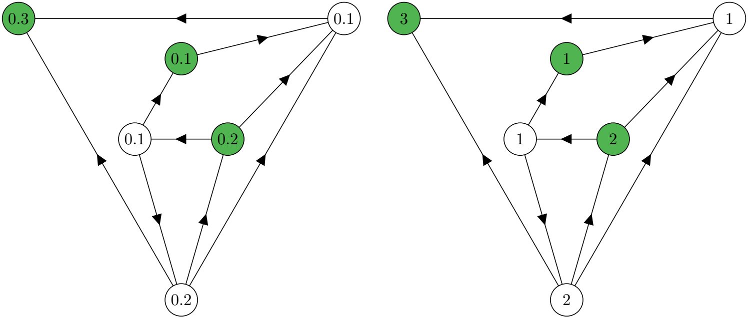
Example: let $M = 10$



Graph bandits: Regret bound of Exp3-IX

$$\sum_{i=1}^{N} \frac{M\hat{p}_{t,i}}{M\hat{p}_{t,i} + \sum_{j \in N_i^-} M\hat{p}_{t,j}} = \sum_{i=1}^{N} \sum_{k \in C_i} \frac{1}{1 + d_k^-} \leq 2\alpha \log\left(1 + \frac{M+N}{\alpha}\right)$$

Example: let $M = 10$



Exp3-IX regret bound

$$R_T \leq \frac{\log N}{\eta} + \left(\frac{\eta}{2} + \gamma\right) \sum_{t=1}^{T} \mathbb{E}\left[2\alpha_t \log\left(1 + \frac{\lceil N^2/\gamma \rceil + N}{\alpha_t}\right) + 2\right]$$

$$R_T = \widetilde{\mathcal{O}}\left(\sqrt{\overline{\alpha}\, T \log(N)}\right)$$

Exp3-IX regret bound

$$R_T \leq \frac{\log N}{\eta} + \left(\frac{\eta}{2} + \gamma\right) \sum_{t=1}^{T} \mathbb{E}\left[2\alpha_t \log\left(1 + \frac{\lceil N^2/\gamma \rceil + N}{\alpha_t}\right) + 2\right]$$

$$R_T = \widetilde{\mathcal{O}}\left(\sqrt{\overline{\alpha} T \log(N)}\right)$$

Next step

Exp3-IX regret bound

$$R_T \leq \frac{\log N}{\eta} + \left(\frac{\eta}{2} + \gamma\right) \sum_{t=1}^{T} \mathbb{E}\left[2\alpha_t \log\left(1 + \frac{\lceil N^2/\gamma \rceil + N}{\alpha_t}\right) + 2\right]$$

$$R_T = \widetilde{\mathcal{O}}\left(\sqrt{\overline{\alpha} T \log(N)}\right)$$

Next step

Generalization of the setting to **combinatorial actions**

Graph bandits: Complex actions

Example: Multiple Ads

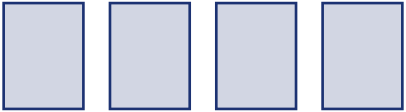
Ad_{01}	Ad_{04}	Ad_{07}	Ad_{10}	Ad_{13}	Ad_{16}
Ad_{02}	Ad_{05}	Ad_{08}	Ad_{11}	Ad_{14}	Ad_{17}
Ad_{03}	Ad_{06}	Ad_{09}	Ad_{12}	Ad_{15}	Ad_{18}

Graph bandits: Complex actions

Example: Multiple Ads

Ad_{01}	Ad_{04}	Ad_{07}	Ad_{10}	Ad_{13}	Ad_{16}
Ad_{02}	Ad_{05}	Ad_{08}	Ad_{11}	Ad_{14}	Ad_{17}
Ad_{03}	Ad_{06}	Ad_{09}	Ad_{12}	Ad_{15}	Ad_{18}

- Display 4 ads (more than 1) and observe losses

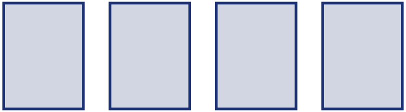


Graph bandits: Complex actions

Example: Multiple Ads

Ad_{01}	Ad_{04}	Ad_{07}	Ad_{10}	Ad_{13}	Ad_{16}
Ad_{02}	Ad_{05}	Ad_{08}	Ad_{11}	Ad_{14}	Ad_{17}
Ad_{03}	Ad_{06}	Ad_{09}	Ad_{12}	Ad_{15}	Ad_{18}

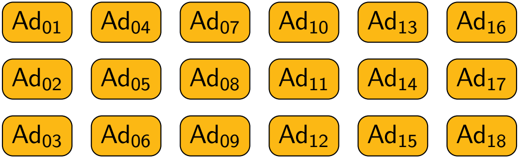
- Display 4 ads (more than 1) and observe losses



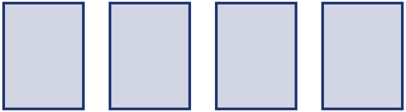
- Play m out of N actions

Graph bandits: Complex actions

Example: Multiple Ads



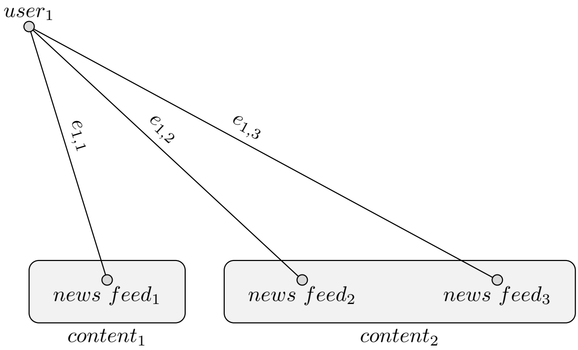
- Display 4 ads (more than 1) and observe losses

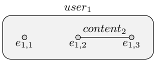


- Play m out of N actions
- Observe losses of all neighbors of played actions

Graph bandits: Complex actions

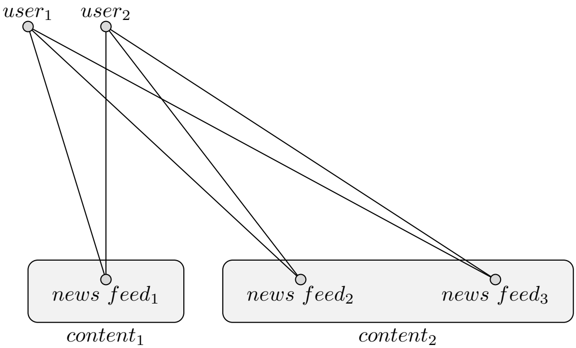
Example: New feeds

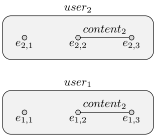




Graph bandits: Complex actions

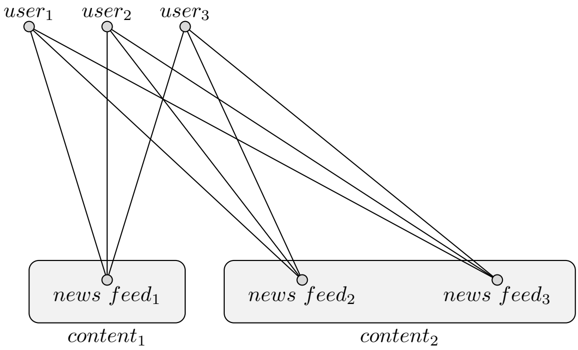
Example: New feeds

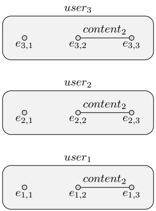




Graph bandits: Complex actions

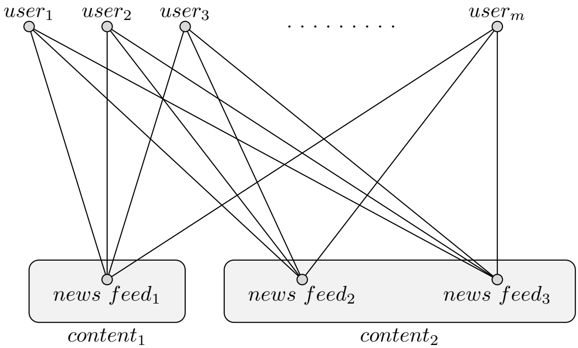
Example: New feeds

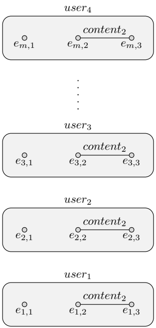




Graph bandits: Complex actions

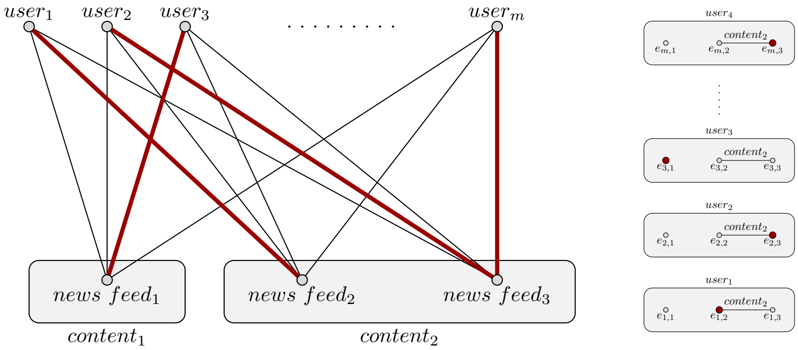
Example: New feeds





Graph bandits: Complex actions

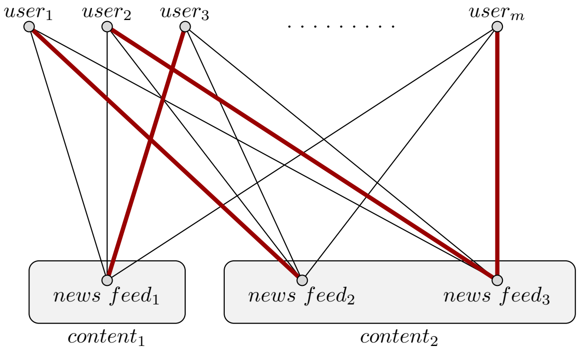
Example: New feeds

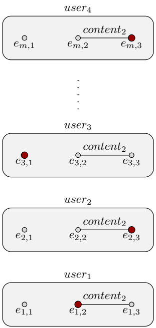


- Play m out of N nodes (combinatorial structure)

Graph bandits: Complex actions

Example: New feeds

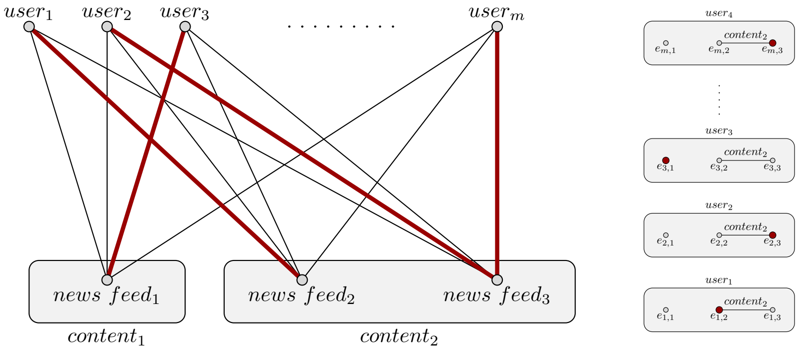




- Play m out of N nodes (combinatorial structure)
- Obtain losses of all played nodes

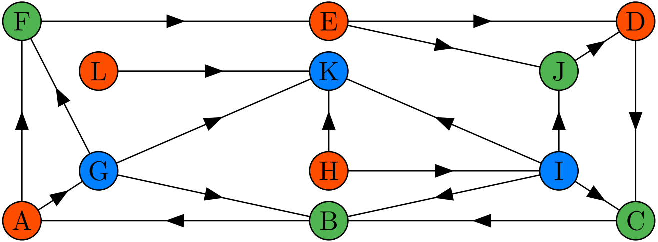
Graph bandits: Complex actions

Example: New feeds



- Play m out of N nodes (combinatorial structure)
- Obtain losses of all played nodes
- Observe losses of all neighbors of played nodes

Graph bandits: Complex actions



- Play action $\mathbf{V}_t \in S \subset \{0,1\}^N$, $\|\mathbf{v}\|_1 \leq m$ from all $\mathbf{v} \in S$
- Obtain losses $\mathbf{V}_t^\mathsf{T} \ell_t$
- Observe additional losses according to the graph

Graph bandits: FPL-IX algorithm

- **Draw perturbation** $Z_{t,i} \sim \mathrm{Exp}(1)$ for all $i \in [N]$

Graph bandits: FPL-IX algorithm

- **Draw perturbation** $Z_{t,i} \sim \mathrm{Exp}(1)$ for all $i \in [N]$

- **Play "the best" action $\mathbf{V}_t$** according to total loss estimate $\widehat{\mathbf{L}}_{t-1}$ and perturbation $\mathbf{Z}_t$

Graph bandits: FPL-IX algorithm

- **Draw perturbation** $Z_{t,i} \sim \text{Exp}(1)$ for all $i \in [N]$

- **Play "the best" action** $\mathbf{V}_t$ according to total loss estimate $\widehat{\mathbf{L}}_{t-1}$ and perturbation $\mathbf{Z}_t$

$$\mathbf{V}_t = \arg\min_{\mathbf{v} \in \mathcal{S}} \mathbf{v}^\mathsf{T} \left(\eta_t \widehat{\mathbf{L}}_{t-1} - \mathbf{Z}_t \right)$$

Graph bandits: FPL-IX algorithm

- **Draw perturbation** $Z_{t,i} \sim \text{Exp}(1)$ for all $i \in [N]$

- **Play "the best" action $\mathbf{V}_t$** according to total loss estimate $\widehat{\mathbf{L}}_{t-1}$ and perturbation $\mathbf{Z}_t$

$$\mathbf{V}_t = \arg\min_{\mathbf{v} \in \mathcal{S}} \mathbf{v}^\mathsf{T} \left(\eta_t \widehat{\mathbf{L}}_{t-1} - \mathbf{Z}_t \right)$$

- **Compute loss estimates**

$$\hat{\ell}_{t,i} = \ell_{t,i} K_{t,i} \mathbb{1}\{\ell_{t,i} \text{ is observed}\}$$

Graph bandits: FPL-IX algorithm

- **Draw perturbation** $Z_{t,i} \sim \text{Exp}(1)$ for all $i \in [N]$

- **Play "the best" action** $\mathbf{V}_t$ according to total loss estimate $\widehat{\mathbf{L}}_{t-1}$ and perturbation $\mathbf{Z}_t$

$$\mathbf{V}_t = \arg\min_{\mathbf{v} \in \mathcal{S}} \mathbf{v}^\top \left(\eta_t \widehat{\mathbf{L}}_{t-1} - \mathbf{Z}_t \right)$$

- **Compute loss estimates**

$$\hat{\ell}_{t,i} = \ell_{t,i} K_{t,i} \mathbb{1}\{\ell_{t,i} \text{ is observed}\}$$

- $K_{t,i}$: geometric random variable with

$$\mathbb{E}\left[K_{t,i}\right] = \frac{1}{o_{t,i} + (1 - o_{t,i})\gamma}$$

Graph bandits: Complex actions

FPL-IX - regret bound

$$R_T = \widetilde{\mathcal{O}}\left(m^{3/2}\sqrt{\sum_{t=1}^{T}\alpha_t}\right) = \widetilde{\mathcal{O}}\left(m^{3/2}\sqrt{\overline{\alpha}T}\right)$$

Graph bandits: Stochastic Rewards

We do better if the losses/rewards are stochastic

Graph bandits: Stochastic Rewards

We do better if the losses/rewards are stochastic

UCB-N - Follow UCB and update the estimates with extra info.

Graph bandits: Stochastic Rewards

We do better if the losses/rewards are stochastic

UCB-N - Follow UCB and update the estimates with extra info.

UCB-MaxN - Follow UCB, but pick the empirically best node in the clique of the node UCB would pick.

Graph bandits: Stochastic Rewards

We do better if the losses/rewards are stochastic

UCB-N - Follow UCB and update the estimates with extra info.

UCB-MaxN - Follow UCB, but pick the empirically best node in the clique of the node UCB would pick.

UCB-LP - linear approximation to the dominating set

Graph bandits: Stochastic Rewards

We do better if the losses/rewards are stochastic

UCB-N - Follow UCB and update the estimates with extra info.

UCB-MaxN - Follow UCB, but pick the empirically best node in the clique of the node UCB would pick.

UCB-LP - linear approximation to the dominating set

http://www.auai.org/uai2012/papers/236.pdf

http://newslab.ece.ohio-state.edu/~buccapat/mabSigfinal.pdf

Known bounds in terms of cliques and dominating sets.

Graph bandits: Very hot topic!

Graph bandits: Very hot topic!

Last paper on arxiv: Fri, 20 Mar 2015 17:21:12 GMT

Graph bandits: Very hot topic!

Last paper on arxiv: Fri, 20 Mar 2015 17:21:12 GMT

Extensions: Noga Alon et al. (2015) Beyond bandits

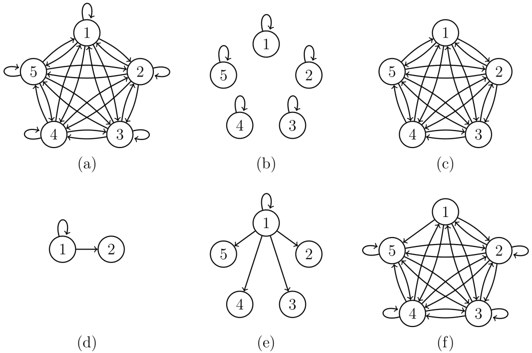


Figure 1: Examples of feedback graphs: (a) *full feedback*, (b) *bandit feedback*, (c) *loopless clique*, (d) *apple tasting*, (e) *revealing action*, (f) a clique minus a self-loop and another edge.

Complete characterization: Bártok et al. (2014)

Graph bandits: Side Observation Summary

- **I**mplicit e**X**ploration idea

Graph bandits: Side Observation Summary

- **I**mplicit e**X**ploration idea
- **Algorithm for simple actions - Exp3-IX**

Graph bandits: Side Observation Summary

- Implicit eXploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea

Graph bandits: Side Observation Summary

- **I**mplicit e**X**ploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea
  - Same regret bound as previous algorithm

Graph bandits: Side Observation Summary

- **I**mplicit e**X**ploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea
  - Same regret bound as previous algorithm
  - No need to know graph before an action is played

Graph bandits: Side Observation Summary

- **I**mplicit e**X**ploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea
  - Same regret bound as previous algorithm
  - No need to know graph before an action is played
  - Computationally efficient

Graph bandits: Side Observation Summary

- **I**mplicit e**X**ploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea
  - Same regret bound as previous algorithm
  - No need to know graph before an action is played
  - Computationally efficient
- **Combinatorial setting with side observations**

Graph bandits: Side Observation Summary

- **I**mplicit e**X**ploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea
  - Same regret bound as previous algorithm
  - No need to know graph before an action is played
  - Computationally efficient
- **Combinatorial setting with side observations**
- **Algorithm for combinatorial setting - FPL-IX**

Graph bandits: Side Observation Summary

- Implicit eXploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea
  - Same regret bound as previous algorithm
  - No need to know graph before an action is played
  - Computationally efficient
- **Combinatorial setting with side observations**
- **Algorithm for combinatorial setting - FPL-IX**
- **Extensions** (open questions)

Graph bandits: Side Observation Summary

- Implicit eXploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea
  - Same regret bound as previous algorithm
  - No need to know graph before an action is played
  - Computationally efficient
- **Combinatorial setting with side observations**
- **Algorithm for combinatorial setting - FPL-IX**
- **Extensions** (open questions)
  - No need to know graph after an action is played

Graph bandits: Side Observation Summary

- **I**mplicit e**X**ploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea
  - Same regret bound as previous algorithm
  - No need to know graph before an action is played
  - Computationally efficient
- **Combinatorial setting with side observations**
- **Algorithm for combinatorial setting - FPL-IX**
- **Extensions** (open questions)
  - No need to know graph after an action is played
  - Stochastic side observations - Random graph models

Graph bandits: Side Observation Summary

- **I**mplicit e**X**ploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea
  - Same regret bound as previous algorithm
  - No need to know graph before an action is played
  - Computationally efficient
- **Combinatorial setting with side observations**
- **Algorithm for combinatorial setting - FPL-IX**
- **Extensions** (open questions)
  - No need to know graph after an action is played
  - Stochastic side observations - Random graph models
  - Exploiting the communities

Graph bandits: Side Observation Summary

- **I**mplicit e**X**ploration idea
- **Algorithm for simple actions - Exp3-IX**
  - Using implicit exploration idea
  - Same regret bound as previous algorithm
  - No need to know graph before an action is played
  - Computationally efficient
- **Combinatorial setting with side observations**
- **Algorithm for combinatorial setting - FPL-IX**
- **Extensions** (open questions)
  - No need to know graph after an action is played
  - Stochastic side observations - Random graph models
  - Exploiting the communities
- **Stochastic losses**

SequeL – INRIA Lille

DaSciM, LIX

Michal Valko

michal.valko@inria.fr

sequel.lille.inria.fr

SpectralTS: Analysis sketch

Divide arms into two groups

- $\Delta_i = \mathbf{b}_*^\mathsf{T}\boldsymbol{\mu} - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu} \leq g\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}$ arm i is **unsaturated**
- $\Delta_i = \mathbf{b}_*^\mathsf{T}\boldsymbol{\mu} - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu} > g\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}$ arm i is **saturated**

SpectralTS: Analysis sketch

Divide arms into two groups

- $\Delta_i = \mathbf{b}_*^\mathsf{T}\boldsymbol{\mu} - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu} \leq g\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}$ arm i is **unsaturated**
- $\Delta_i = \mathbf{b}_*^\mathsf{T}\boldsymbol{\mu} - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu} > g\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}$ arm i is **saturated**

Saturated arm

- Small standard deviation $\rightarrow$ accurate regret estimate.
- **High regret** on playing the arm $\rightarrow$ **Low probability** of picking

SpectralTS: Analysis sketch

Divide arms into two groups

- $\Delta_i = \mathbf{b}_*^\top \boldsymbol{\mu} - \mathbf{b}_i^\top \boldsymbol{\mu} \leq g \|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}$ arm i is **unsaturated**
- $\Delta_i = \mathbf{b}_*^\top \boldsymbol{\mu} - \mathbf{b}_i^\top \boldsymbol{\mu} > g \|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}$ arm i is **saturated**

Saturated arm

- Small standard deviation $\rightarrow$ accurate regret estimate.
- **High regret** on playing the arm $\rightarrow$ **Low probability** of picking

Unsaturated arm

- **Low regret** bounded by a factor of standard deviation
- **High probability** of picking

SpectralTS: Analysis sketch

- Confidence ellipsoid for estimate $\hat{\boldsymbol{\mu}}$ of $\boldsymbol{\mu}$ (with probability $1 - \delta/T^2$)
  - Using analysis of OFUL algorithm (Abbasi-Yadkori et al., 2011)

$$|\mathbf{b}_i^\mathsf{T}\hat{\boldsymbol{\mu}} - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu}| \leq \left(R\sqrt{2\log\left(\frac{|\mathbf{B}_T|^{1/2}T^2}{|\boldsymbol{\Lambda}|^{1/2}\delta}\right)} + C\right)\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}$$

SpectralTS: Analysis sketch

- Confidence ellipsoid for estimate $\hat{\boldsymbol{\mu}}$ of $\boldsymbol{\mu}$ (with probability $1 - \delta/T^2$)
  - Using analysis of OFUL algorithm (Abbasi-Yadkori et al., 2011)

$$|\mathbf{b}_i^\mathsf{T}\hat{\boldsymbol{\mu}} - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu}| \leq \left(R\sqrt{2\log\left(\frac{|\mathbf{B}_T|^{1/2}T^2}{|\mathbf{\Lambda}|^{1/2}\delta}\right)} + C\right)\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}$$

- The key result coming from spectral properties of $\mathbf{B}_t$.

$$\log\frac{|\mathbf{B}_t|}{|\mathbf{\Lambda}|} \leq 2d\log\left(1 + \frac{T}{\lambda}\right)$$

SpectralTS: Analysis sketch

- Confidence ellipsoid for estimate $\hat{\boldsymbol{\mu}}$ of $\boldsymbol{\mu}$ (with probability $1 - \delta/T^2$)
  - Using analysis of OFUL algorithm (Abbasi-Yadkori et al., 2011)

$$|\mathbf{b}_i^\mathsf{T}\hat{\boldsymbol{\mu}} - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu}| \leq \left(R\sqrt{2\,d\log\left(\frac{(\lambda+T)T^2}{\delta\lambda}\right)} + C\right)\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}} = \ell\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}$$

- The key result coming from spectral properties of $\mathbf{B}_t$.

$$\log\frac{|\mathbf{B}_t|}{|\boldsymbol{\Lambda}|} \leq 2d\log\left(1 + \frac{T}{\lambda}\right)$$

SpectralTS: Analysis sketch

- Confidence ellipsoid for estimate $\hat{\boldsymbol{\mu}}$ of $\boldsymbol{\mu}$ (with probability $1 - \delta/T^2$)
  - Using analysis of OFUL algorithm (Abbasi-Yadkori et al., 2011)

$$|\mathbf{b}_i^\mathsf{T}\hat{\boldsymbol{\mu}} - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu}| \le \left(R\sqrt{2\,d\log\left(\frac{(\lambda+T)T^2}{\delta\lambda}\right)} + C\right)\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}} = \ell\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}$$

- The key result coming from spectral properties of $\mathbf{B}_t$.

$$\log\frac{|\mathbf{B}_t|}{|\boldsymbol{\Lambda}|} \le 2d\log\left(1 + \frac{T}{\lambda}\right)$$

- Concentration of sample $\tilde{\boldsymbol{\mu}}$ around mean $\hat{\boldsymbol{\mu}}$ (with probability $1 - 1/T^2$)
  - Using concentration inequality for Gaussian random variable.

$$|\mathbf{b}_i^\mathsf{T}\tilde{\boldsymbol{\mu}} - \mathbf{b}_i^\mathsf{T}\hat{\boldsymbol{\mu}}| \le \left(R\sqrt{6d\log\left(\frac{\lambda+T}{\delta\lambda}\right)} + C\right)\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}\sqrt{4\log(TN)} = v\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}\sqrt{4\log(TN)}$$

SpectralTS: Analysis sketch

Define $\text{regret}'(t) = \text{regret}(t) \cdot \mathbb{1}\{|\mathbf{b}_i^\mathsf{T}\hat{\boldsymbol{\mu}}(t) - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu}| \leq \ell \|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}\}$

$$\text{regret}'(t) \leq \frac{11g}{p}\|\mathbf{b}_{a(t)}\|_{\mathbf{B}_t^{-1}} + \frac{1}{T^2}$$

SpectralTS: Analysis sketch

Define $\text{regret}'(t) = \text{regret}(t) \cdot \mathbb{1}\{|\mathbf{b}_i^\mathsf{T}\hat{\boldsymbol{\mu}}(t) - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu}| \leq \ell\|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}\}$

$$\text{regret}'(t) \leq \frac{11g}{p}\|\mathbf{b}_{a(t)}\|_{\mathbf{B}_t^{-1}} + \frac{1}{T^2}$$

Super-martingale (i.e. $\mathbb{E}[Y_t - Y_{t-1}|\mathcal{F}_{t-1}] \leq 0$)

$$X_t = \text{regret}'(t) - \frac{11g}{p}\|\mathbf{b}_{a(t)}\|_{\mathbf{B}_t^{-1}} - \frac{1}{T^2}$$

$$Y_t = \sum_{w=1}^{t} X_w.$$

$(Y_t;\ t = 0, \dots, T)$ is a **super-martingale** process w.r.t. history $\mathcal{F}_t$.

SpectralTS: Analysis sketch

Define $\text{regret}'(t) = \text{regret}(t) \cdot \mathbb{1}\{|\mathbf{b}_i^\mathsf{T}\hat{\boldsymbol{\mu}}(t) - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu}| \leq \ell \|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}\}$

$$\text{regret}'(t) \leq \frac{11g}{p}\|\mathbf{b}_{a(t)}\|_{\mathbf{B}_t^{-1}} + \frac{1}{T^2}$$

Super-martingale (i.e. $\mathbb{E}[Y_t - Y_{t-1}|\mathcal{F}_{t-1}] \leq 0$)

$$\begin{aligned} X_t &= \text{regret}'(t) - \frac{11g}{p}\|\mathbf{b}_{a(t)}\|_{\mathbf{B}_t^{-1}} - \frac{1}{T^2} \\ Y_t &= \sum_{w=1}^{t} X_w. \end{aligned}$$

$(Y_t;\ t = 0, \ldots, T)$ is a **super-martingale** process w.r.t. history $\mathcal{F}_t$.

Azuma-Hoeffding inequality for super-martingale, w. p. $1 - \delta/2$:

$$\sum_{t=1}^{T} \text{regret}'(t) \leq \frac{11g}{p}\sum_{t=1}^{T}\|\mathbf{b}_{a(t)}\|_{\mathbf{B}_t^{-1}} + \frac{1}{T} + \frac{g}{p}\left(\frac{11}{\sqrt{\lambda}} + 2\right)\sqrt{2T\ln\frac{2}{\delta}}$$

SpectralTS: Analysis sketch

Define $\text{regret}'(t) = \text{regret}(t) \cdot \mathbb{1}\{|\mathbf{b}_i^\mathsf{T}\hat{\boldsymbol{\mu}}(t) - \mathbf{b}_i^\mathsf{T}\boldsymbol{\mu}| \leq \ell \|\mathbf{b}_i\|_{\mathbf{B}_t^{-1}}\}$

$$\text{regret}'(t) \leq \frac{11g}{p}\|\mathbf{b}_{a(t)}\|_{\mathbf{B}_t^{-1}} + \frac{1}{T^2}$$

Super-martingale (i.e. $\mathbb{E}[Y_t - Y_{t-1}|\mathcal{F}_{t-1}] \leq 0$)

$$X_t = \text{regret}'(t) - \frac{11g}{p}\|\mathbf{b}_{a(t)}\|_{\mathbf{B}_t^{-1}} - \frac{1}{T^2}$$

$$Y_t = \sum_{w=1}^{t} X_w.$$

$(Y_t;\ t = 0, \ldots, T)$ is a **super-martingale** process w.r.t. history $\mathcal{F}_t$.

Azuma-Hoeffding inequality for super-martingale, w. p. $1 - \delta/2$:

$$\sum_{t=1}^{T} \text{regret}'(t) \leq \frac{11g}{p}\sum_{t=1}^{T}\|\mathbf{b}_{a(t)}\|_{\mathbf{B}_t^{-1}} + \frac{1}{T} + \frac{g}{p}\left(\frac{11}{\sqrt{\lambda}} + 2\right)\sqrt{2T\ln\frac{2}{\delta}}$$

SpectralEliminator: Analysis

SpectralEliminator

- Divide time into sets ($t_1 = 1 \leq t_2 \leq \dots$) to introduce independence for Azuma-Hoeffding inequality and observe $R_T \leq \sum_{j=0}^{J}(t_{j+1} - t_j)\left[\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j \rangle + (\|\mathbf{x}^*\|_{\mathbf{V}_j^{-1}} + \|\mathbf{x}_t\|_{\mathbf{V}_j^{-1}})\beta\right]$

SpectralEliminator: Analysis

SpectralEliminator

- Divide time into sets $(t_1 = 1 \leq t_2 \leq \ldots)$ to introduce independence for Azuma-Hoeffding inequality and observe $R_T \leq \sum_{j=0}^{J}(t_{j+1} - t_j)\left[\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle + (\|\mathbf{x}^*\|_{\mathbf{V}_j^{-1}} + \|\mathbf{x}_t\|_{\mathbf{V}_j^{-1}})\beta\right]$
- Bound $\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle$ for each phase

SpectralEliminator: Analysis

SpectralEliminator

- Divide time into sets $(t_1 = 1 \leq t_2 \leq \dots)$ to introduce independence for Azuma-Hoeffding inequality and observe $R_T \leq \sum_{j=0}^{J}(t_{j+1} - t_j)\left[\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle + (\|\mathbf{x}^*\|_{\mathbf{V}_j^{-1}} + \|\mathbf{x}_t\|_{\mathbf{V}_j^{-1}})\beta\right]$
- Bound $\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle$ for each phase
- No bad arms: $\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle \leq (\|\mathbf{x}^*\|_{\mathbf{V}_j^{-1}} + \|\mathbf{x}_t\|_{\mathbf{V}_j^{-1}})\beta$

SpectralEliminator: Analysis

SpectralEliminator

- Divide time into sets $(t_1 = 1 \leq t_2 \leq \dots)$ to introduce independence for Azuma-Hoeffding inequality and observe $R_T \leq \sum_{j=0}^{J}(t_{j+1} - t_j)\left[\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle + (\|\mathbf{x}^*\|_{\mathbf{V}_j^{-1}} + \|\mathbf{x}_t\|_{\mathbf{V}_j^{-1}})\beta\right]$
- Bound $\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle$ for each phase
- No bad arms: $\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle \leq (\|\mathbf{x}^*\|_{\mathbf{V}_j^{-1}} + \|\mathbf{x}_t\|_{\mathbf{V}_j^{-1}})\beta$
- By algorithm: $\|\mathbf{x}\|^2_{\mathbf{V}_j^{-1}} \leq \frac{1}{t_j - t_{j-1}} \sum_{s=t_{j-1}+1}^{t_j} \|\mathbf{x}_s\|^2_{\mathbf{V}_{s-1}^{-1}}$

SpectralEliminator: Analysis

SpectralEliminator

- Divide time into sets $(t_1 = 1 \leq t_2 \leq \dots)$ to introduce independence for Azuma-Hoeffding inequality and observe $R_T \leq \sum_{j=0}^{J}(t_{j+1} - t_j)\big[\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle + (\|\mathbf{x}^*\|_{\mathbf{V}_j^{-1}} + \|\mathbf{x}_t\|_{\mathbf{V}_j^{-1}})\beta\big]$
- Bound $\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle$ for each phase
- No bad arms: $\langle \mathbf{x}^* - \mathbf{x}_t, \hat{\alpha}_j\rangle \leq (\|\mathbf{x}^*\|_{\mathbf{V}_j^{-1}} + \|\mathbf{x}_t\|_{\mathbf{V}_j^{-1}})\beta$
- By algorithm: $\|\mathbf{x}\|^2_{\mathbf{V}_j^{-1}} \leq \frac{1}{t_j - t_{j-1}} \sum_{s=t_{j-1}+1}^{t_j} \|\mathbf{x}_s\|^2_{\mathbf{V}_{s-1}^{-1}}$
- $\sum_{s=t_{j-1}+1}^{t_j} \min\left(1, \|\mathbf{x}_s\|^2_{\mathbf{V}_{s-1}^{-1}}\right) \leq \log \frac{|\mathbf{V}_j|}{|\mathbf{\Lambda}|}$