POLITECNICO
MILANO 1863

ONLINE LEARNING APPLICATIONS

Professor: Matteo Castiglioni

Teaching assistant: Matteo Bollini

Group members:
Alberte Baht
Alessandro Giovoni
Marco Minaudo
Tommaso Giovanni Volonteri

GENERAL SETTING

At each round $t \in \{1,2,\dots,T\}$:

1. A company chooses the prices $(p_i)^N{}_{i=1}$ of N different types of products from a set of possible prices $P = \{0.1,0.2,\dots,1,1.1\}$ (an arm that cannot win the selling is always present in P)

1. A buyer with an unknown valuation v_i for each type of product arrives

2. The buyer buys a unit of the i-th product if $p_i \leq v_i$

REMARK: INVENTORY CONSTRAINT

The company can produce a total number of products B

REQUIREMENT 1

Setting:

- Stochastic environment
- $N = 1$ and its production cost c

TASK 1

Build a pricing strategy using UCB1 ignoring the inventory constraint

TASK 2

Build a pricing strategy using UCB1 to handle the inventory constraint

REQUIREMENT 1

Setting:

- Stochastic environment
- $N = 1$ and its production cost c

TASK 1

Build a pricing strategy using UCB1 ignoring the inventory constraint

TASK 2

Build a pricing strategy using UCB1 to handle the inventory constraint

DEFINITIONS

Number of pulls of the price p

$$N_{t-1}(p) = \sum_{t'=1}^{t-1} \mathbb{I}_{p_{t'}=p}$$

Selling result with price p at round t:

$$s_t(p) = \mathbb{I}_{p \le v}$$

Utility of the price p at round t

$$f_t(p) = (p_t - c)s_t$$

REQUIREMENT 1.1

ALGORITHM 1.1

INPUTS: T, P

For $t = 1,2, \dots T$:

- Compute $\overline{f_t(p)} = \frac{1}{N_{t-1}(p)} \sum_{t'=1}^{t-1} f_{t'}(p) \mathbb{I}_{p_{t'}=p} \quad \forall p \in P$
- Define $f_t^{UCB}(p) = \overline{f_t(p)} + \sqrt{\frac{2\log(T)}{N_{t-1}(p)}} \quad \forall p \in P$
- Set $p_t = argmax_{p \in P} \; f_t^{UCB}(p)$
- Observe $s_t(p_t)$ and $f_t(p_t)$

IMPLEMENTATION

- Valuations are $\mathcal{U}(0,1)$ - distributed
  - **Class Buyer:**
    - *round*
    - *update*
  - **Class Seller**
    - *pull_arm*
    - *update*

REQUIREMENT 1.1
Results

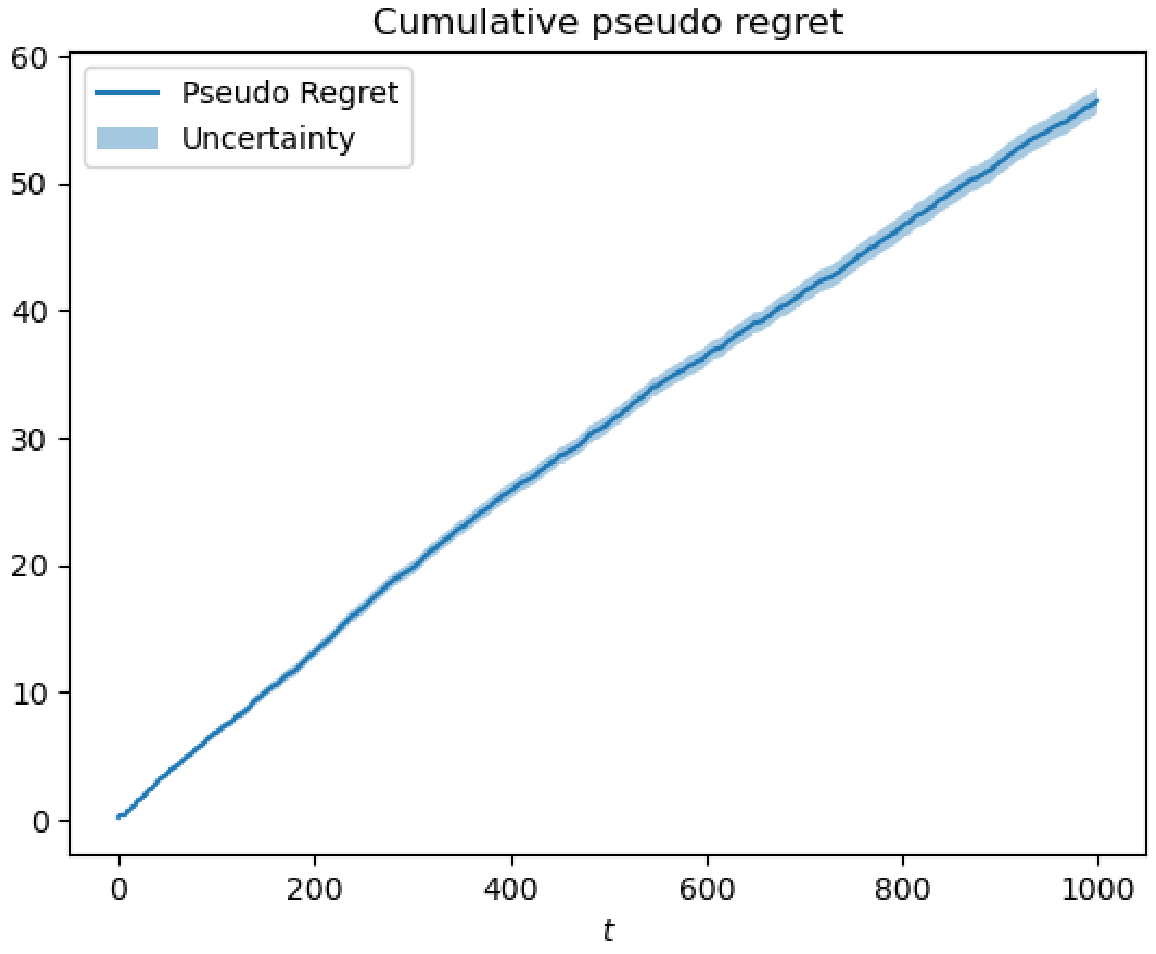
FRAMEWORK

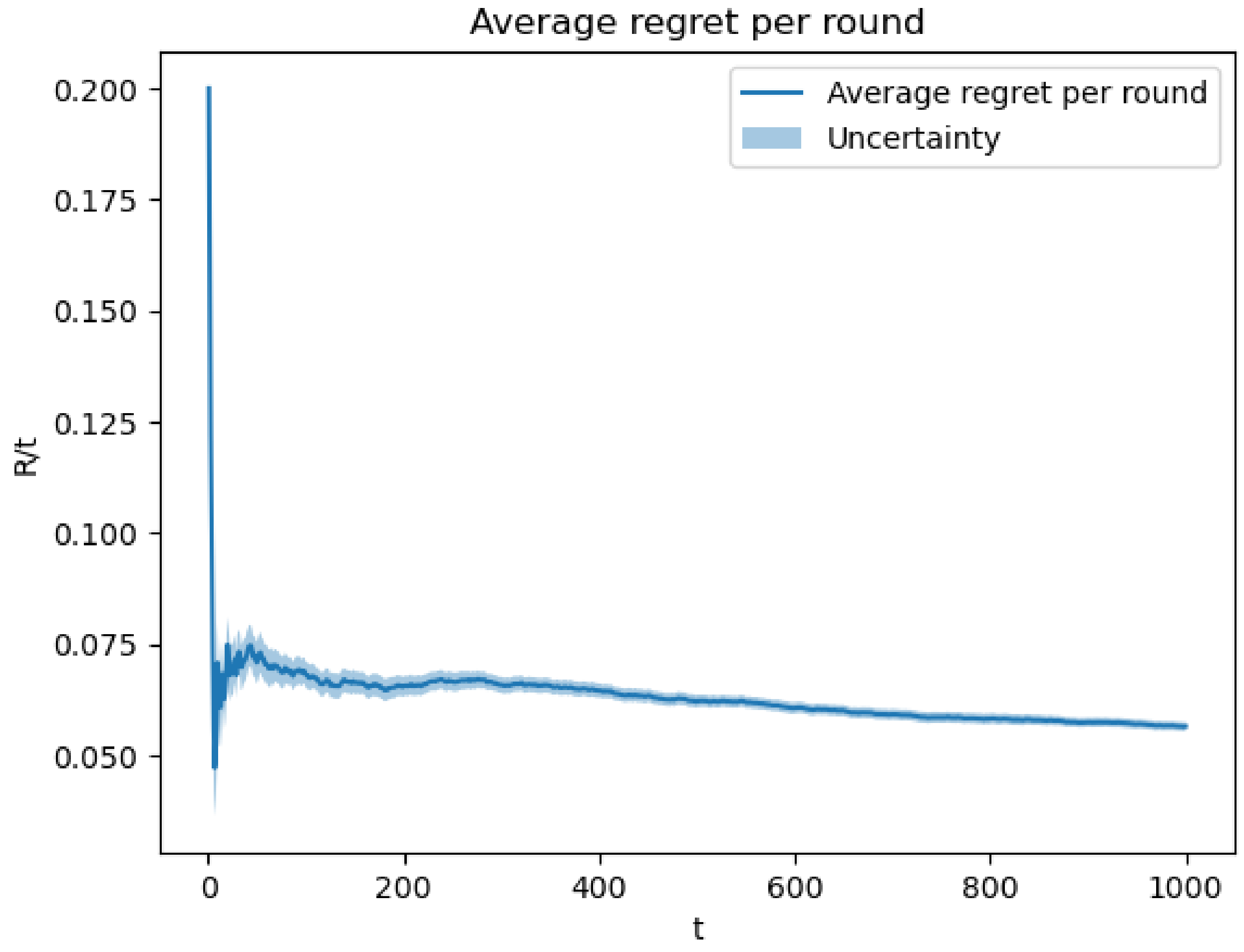
$T = 1000$ $P = \{0.1, 0.2, \ldots, 1, 1.1\}$ $c = 0.1$ $n_{trails} = 50$

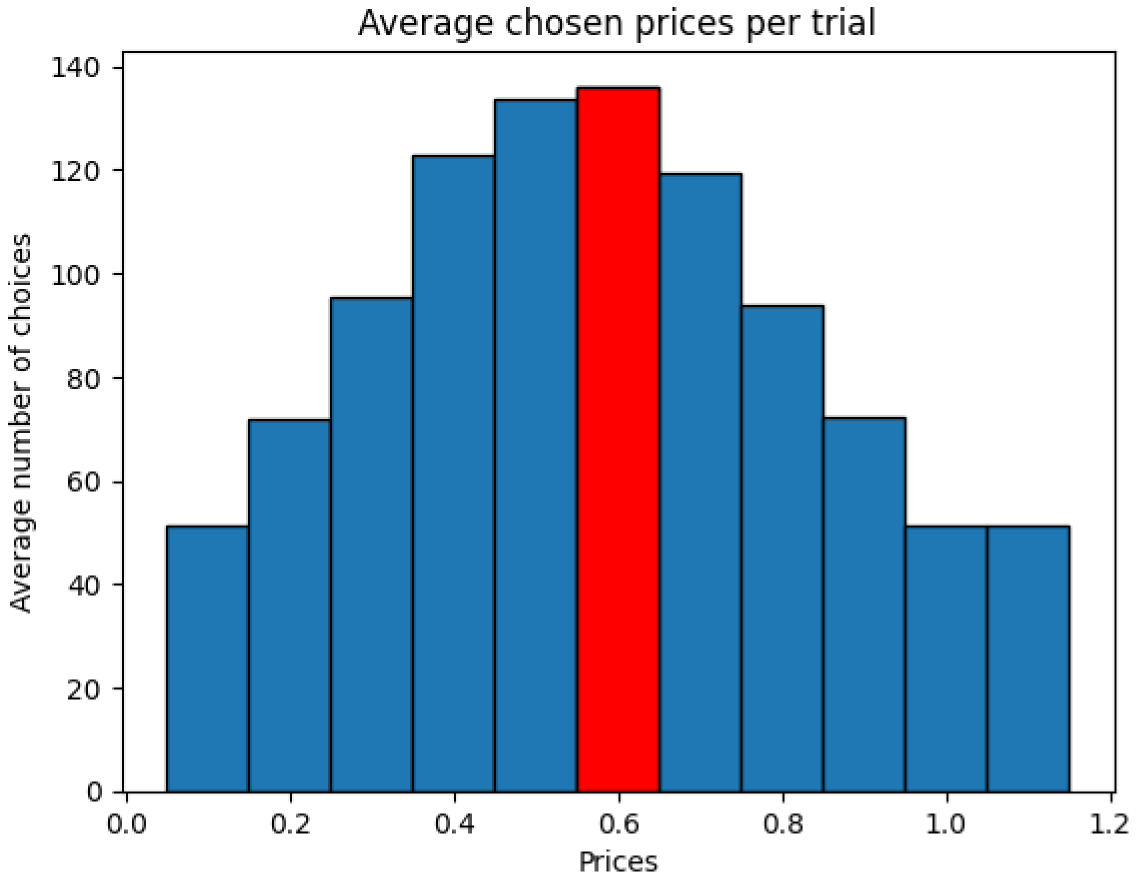
CLAIRVOYANT

The best expected price to set:

$$p^* = argmax_P \mathbb{E}[f(p)]$$







REQUIREMENT 1

Setting:

- Stochastic environment
- $N = 1$ and its production cost c

TASK 1

Build a pricing strategy using UCB1 ignoring the inventory constraint

TASK 2

Build a pricing strategy using UCB1 to handle the inventory constraint

REQUIREMENT 1.2

ALGORITHM 1.2

INPUTS: $T, P, B, \rho = \frac{B}{T}$

For $t = 1,2, \dots T$:

- Compute $\overline{f_t(p)}$ and $f_t^{UCB}(p)$ $\forall p \in P$
- Compute $\overline{s_t(p)}$ and $s_t^{LCB}(p)$ $\forall p \in P$
- Retrieve γ_t solving $LP(f_t^{UCB}, s_t^{LCB}, P)$ and sample $p_t \sim \gamma_t$
- Observe $s_t(p_t)$ and $f_t(p_t)$
- $B = B - s_t(p_t) \rightarrow$ stop if $B < 1$

IMPLEMENTATION

- Valuations are $\mathcal{U}(0,1)$ - distributed
- Definition of **Buyer** and **Seller**
- $LP\ (f, s, A)$ is solved using ***linprog*** from ***scipy.optimize*** library

$$
LP(f,s,A) \begin{cases} \max\limits_{\gamma \in \mathbb{R}^K} \sum\limits_p \gamma(p)\, f(p) \\ \text{s.t.} \quad \sum\limits_p \gamma(p)\, c(p) \le \rho = \frac{B}{T} \\ \sum\limits_{j=1}^{|A|} \gamma_j = 1 \\ 0 \le \gamma_j \le 1 \ \ \forall j = 1 \dots |A| \end{cases}
$$

REQUIREMENT 1.2
Results

FRAMEWORK

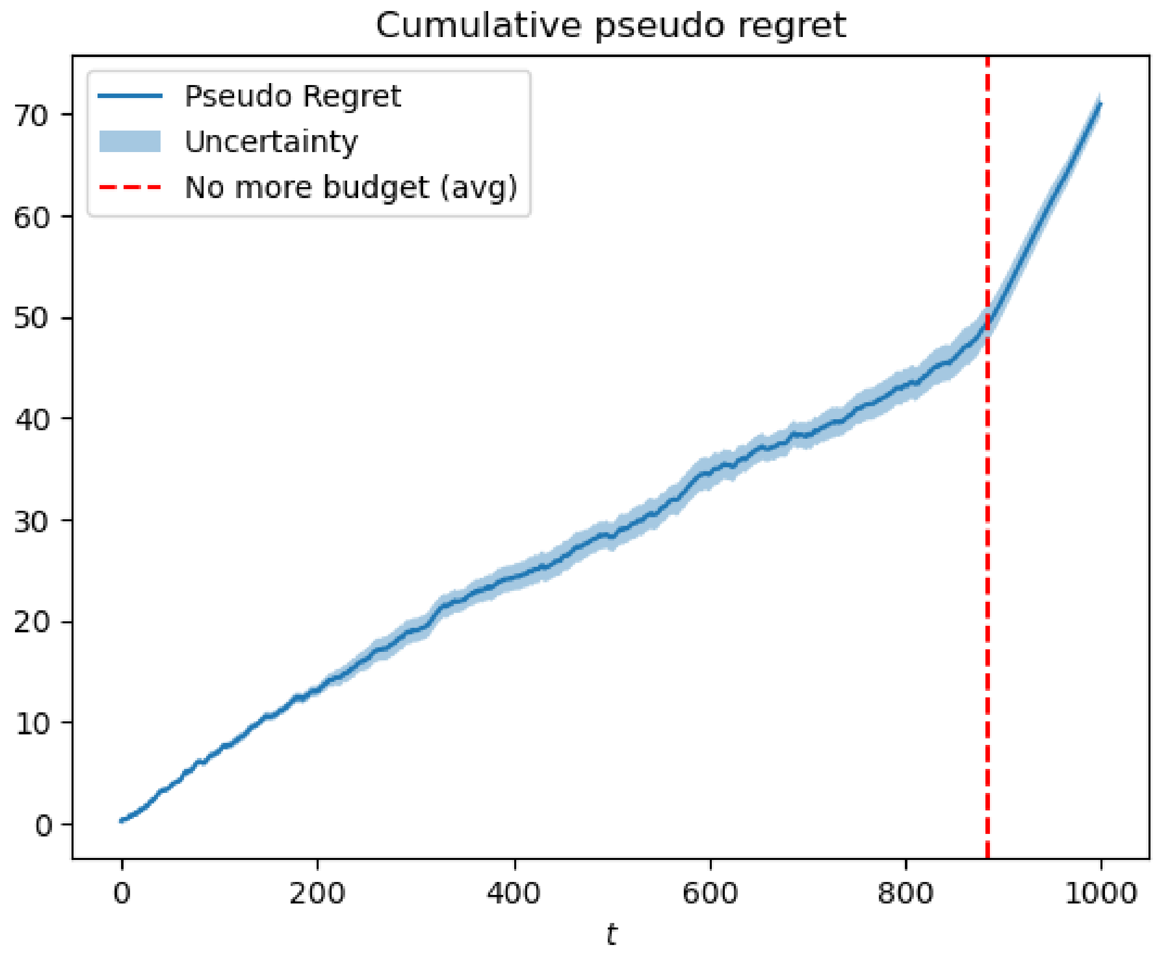
$T = 1000$ $B = 400$ $P = \{0.1, 0.2, \ldots, 1, 1.1\}$ $c = 0.1$ $n_{trails} = 10$

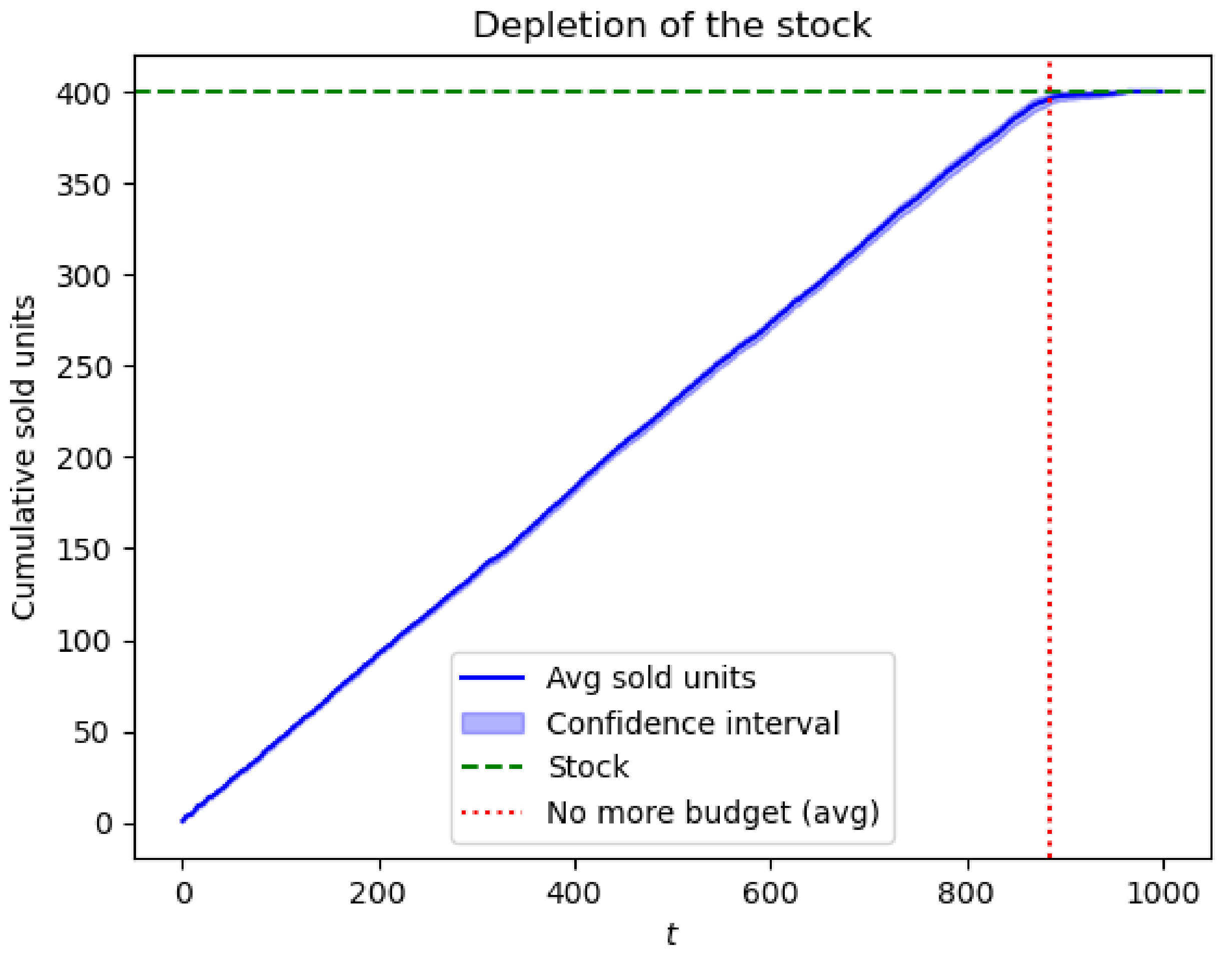
CLAIRVOYANT

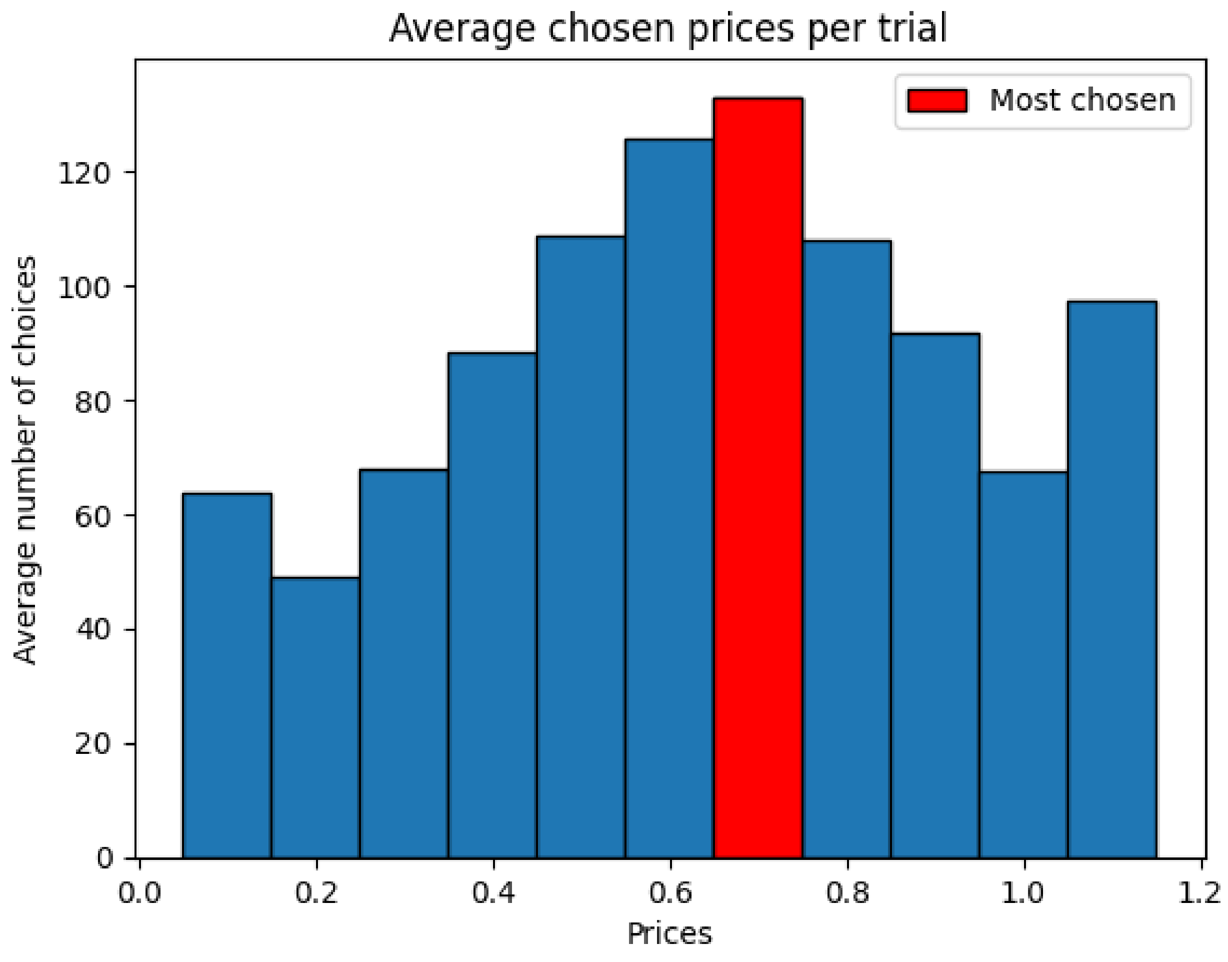
At each round it gains the best fixed expected utility:

$$f^*(\gamma) = \sum_p \gamma(p)(p - c)Pr(p < v)$$

$\gamma(p)$ founded by $LP(f^*(\gamma), s^*(\gamma), P)$







REQUIREMENT 2

Setting:

- Stochastic environment
- $N > 1$ and their production costs $\boldsymbol{c} = (c_i)^N{}_{i=1}$
- $\boldsymbol{P}$ set of superarms

TASK

Build a pricing strategy using Combinatorial-UCB with the inventory constraint.

REQUIREMENT 2

Setting:

- Stochastic environment
- $N > 1$ and their production costs $\boldsymbol{c} = (c_i)^N{}_{i=1}$
- $\boldsymbol{P}$ set of superarms

TASK

Build a pricing strategy using Combinatorial-UCB with the inventory constraint.

DEFINITIONS

Number of pulls of the price p for item i

$$N_{t-1}(p^{(i)}) = \sum_{t'=1}^{t-1} \mathbb{I}_{p_{t'}^{(i)}=p^{(i)}}$$

Selling result with superarm $\boldsymbol{p}$ at round t:

$$S_t(\boldsymbol{p}) = \sum_{i=1}^{N} \mathbb{I}_{p^{(i)} \leq v_i}$$

Utility of the superarm $\boldsymbol{p}$ at round t

$$F_t(\boldsymbol{p}) = \sum_{i=1}^{N} \left(p^{(i)} - c_i\right) s_t(p^{(i)})$$

REQUIREMENT 2

ALGORITHM 2

INPUTS: $T, P, N, B, \rho = \frac{B}{T}$

For $t = 1,2, \dots T$:

- Compute $F_t^{UCB}(\boldsymbol{p}) = \sum_{i=1}^{N} f_t^{UCB}(p^{(i)}) \quad \forall \boldsymbol{p} \in \boldsymbol{P}$
- Compute $S_t^{LCB}(\boldsymbol{p}) = \sum_{i=1}^{N} s_t^{LCB}(p^{(i)}) \quad \forall \boldsymbol{p} \in \boldsymbol{P}$
- Retrieve $\boldsymbol{\gamma}_t$ solving $LP(F_t^{UCB}, S_t^{LCB}, \boldsymbol{P})$ and sample $\boldsymbol{p}_t \sim \boldsymbol{\gamma}_t$
- Observe $S_t(\boldsymbol{p}_t)$ and $F_t(\boldsymbol{p}_t)$
- Decrease the inventory: $B = B - S_t(\boldsymbol{p}_t)$
- $B = B - s_t(p_t) \rightarrow$ stop if $B < N$

IMPLEMENTATION

- Valuations are $\mathcal{U}$ – distributed with different means among the products (**independece assumption**)
- Definition of **Buyer** and **Seller**
- List of superarms generated using ***itertools*** library
- Parallelization on trials using ***Parallel(n_jobs=-1)*** and ***delayed*** from ***joblib*** library

REQUIREMENT 2
Results

FRAMEWORK

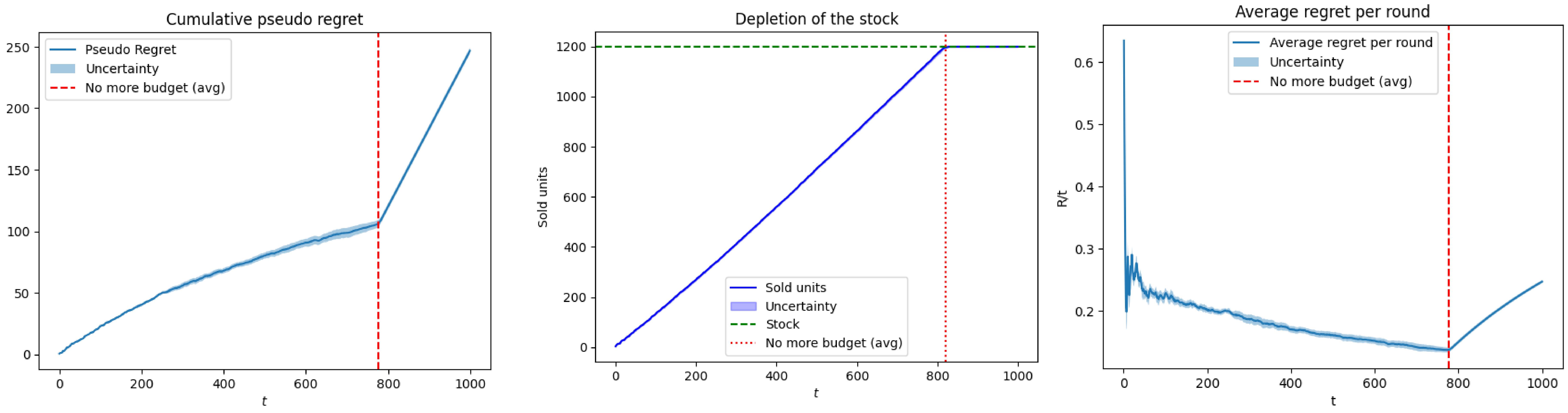
$N = 3 \quad T = 1000 \quad B = 1200 \quad P = \{0.1, 0.2, \dots, 1, 1.1\} \quad c_i = 0.1 \; \forall i \quad n_{trails} = 5$

CLAIRVOYANT

At each round it gains the best fixed expected utility:

$$F^*(\boldsymbol{\gamma}) = \sum_{\boldsymbol{p}} \gamma(\boldsymbol{p}) \sum_{i=1}^{N} (p^{(i)} - c^{(i)}) Pr(p^{(i)} < v^{(i)})$$

$\boldsymbol{\gamma}(\boldsymbol{p})$ founded by $LP(F^*(\boldsymbol{\gamma}), S^*(\boldsymbol{\gamma}), \boldsymbol{P})$



REQUIREMENT 3

TASK

Design a primal-dual algorithm to simulate sellings of a single product with the inventory constraint

Setting 1:

- Stochastic environment
- $N = 1$ and its production cost c

Setting 2:

- Highly non-stationary environment
- $N = 1$ and its production cost c

REQUIREMENT 3

ALGORITHM 3

INPUTS: T, P, N, B

INITIALIZATION: $\lambda = (1, \dots, 1),\ \ \rho = \frac{B}{T},\ \eta = T^{-\frac{1}{2}}$

For $t = 1,2, \dots T$:

- Sample p_t from a regret minimizer (**Hedge**)
- Observe $f_t(p)$ and $s_t(p)\ \ \forall p \in P$ (Full feedback)
- Compute: $L_t(p) = f_t(p) - \lambda_t(p)(s_t(p) - \rho\ )\ \forall p \in P$
- $\lambda_{t+1}(p) = \Pi_{\left[0,\frac{1}{\rho}\right]}(\lambda_t(p) - \eta\big(\rho - s_t(p)\big)\ \ \forall p \in P$
- Update Hedge passing $l_t(\mathrm{p}) = 1 - L_{t,norm}(p)\ \forall p \in P$
- $B = B - s_t(p_t) \rightarrow$ stop if $B < 1$

NORMALIZATION OF THE LAGRANGIAN

$$L_{norm} = min\left(1, max\left(0, \frac{L(p) - L_{min}}{L_{MAX} - L_{min}}\right)\right)$$

Where:

- $L_{MAX} = (max_{p \in P \setminus \{\underset{p}{max}(p)\}}(p) - c) + max\big(\lambda\rho, \lambda(1 - \rho)\big)$
- $L_{min} = min(\lambda\rho, \lambda(1 - \rho))$

REQUIREMENT 3

IMPLEMENTATION: SETTING 1

- Valuations are $\mathcal{U}(0,1)$ - distributed
- Definition of **Buyer, Hedge Agent** and **MultiplicativePacingSeller**
- **Clairvoyant** : same of Requirement 1.2
  - Computation of the best distribution solving $LP(f^*(\gamma), s^*(\gamma), P)$ problem
  - Expected utility

IMPLEMENTATION: SETTING 2

- Generation of valuations by partitioning $[T]$ in blocks :
  - Each block consits of R rounds
  - In odd blocks: $\mathcal{U}$- distributed, with a changing mean
  - In even blocks: $Beta(\alpha, \alpha)$-distributed with a changing α
  - Generated using ***beta*** from ***scipy.stats***
- Definition of **Buyer**, **Hedge Agent** and **Seller**
- **Clairvoyant**:
  At each round it gains the best fixed expected utility:

$$f^*(\gamma) = \sum_p \gamma(p)(p - c)\, Pr_{emp}\,(p < v)$$

Where:

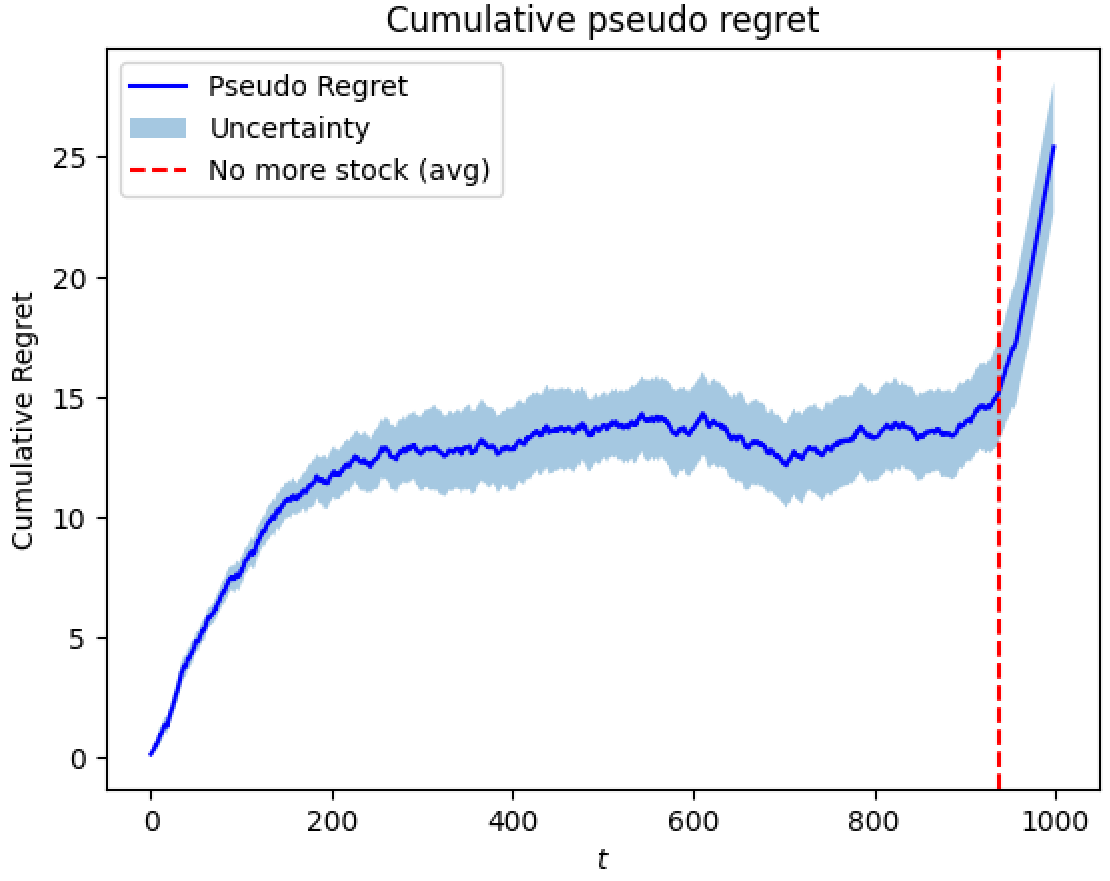
$Pr_{emp} =$ empirical probability $= \frac{\#\, of \text{ successes}}{T}$

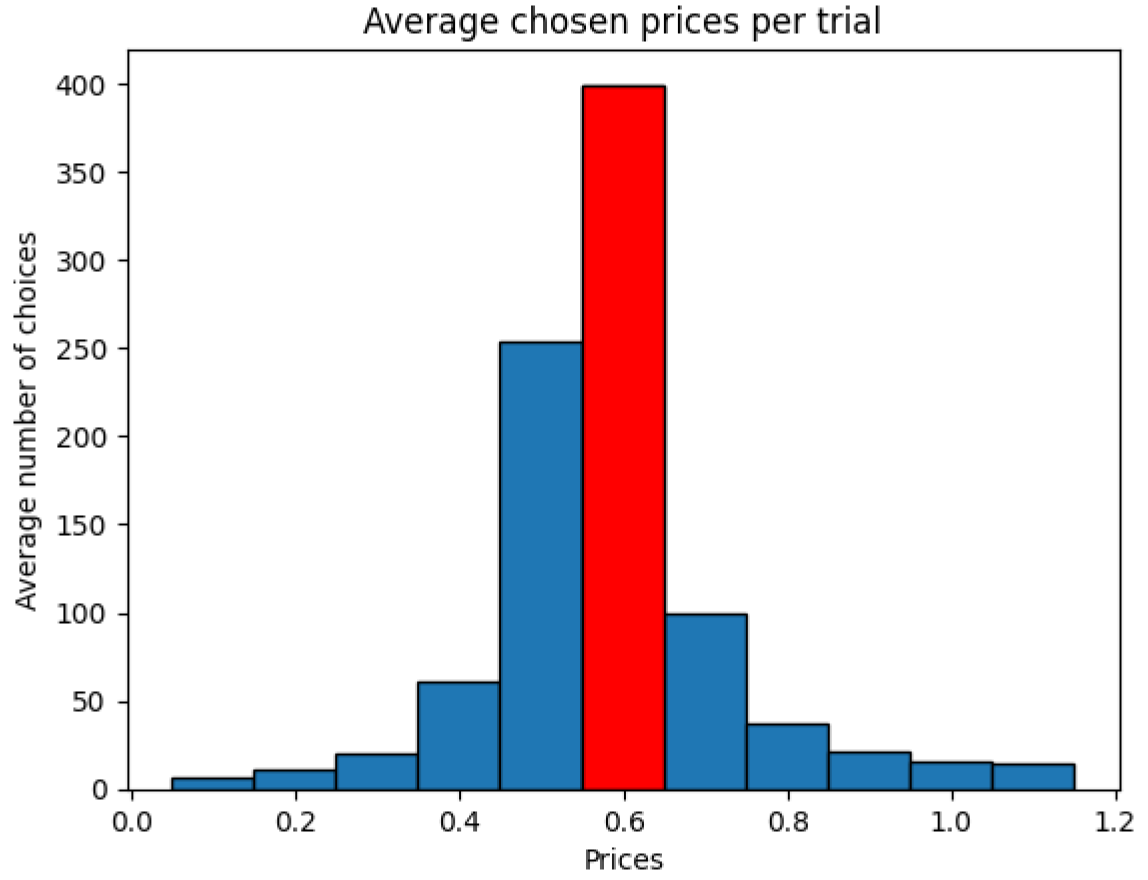
$\gamma(p)$ founded by $LP(f^*(\gamma), s^*(\gamma), P)$

REQUIREMENT 3
Results

FRAMEWORK: SETTING 1

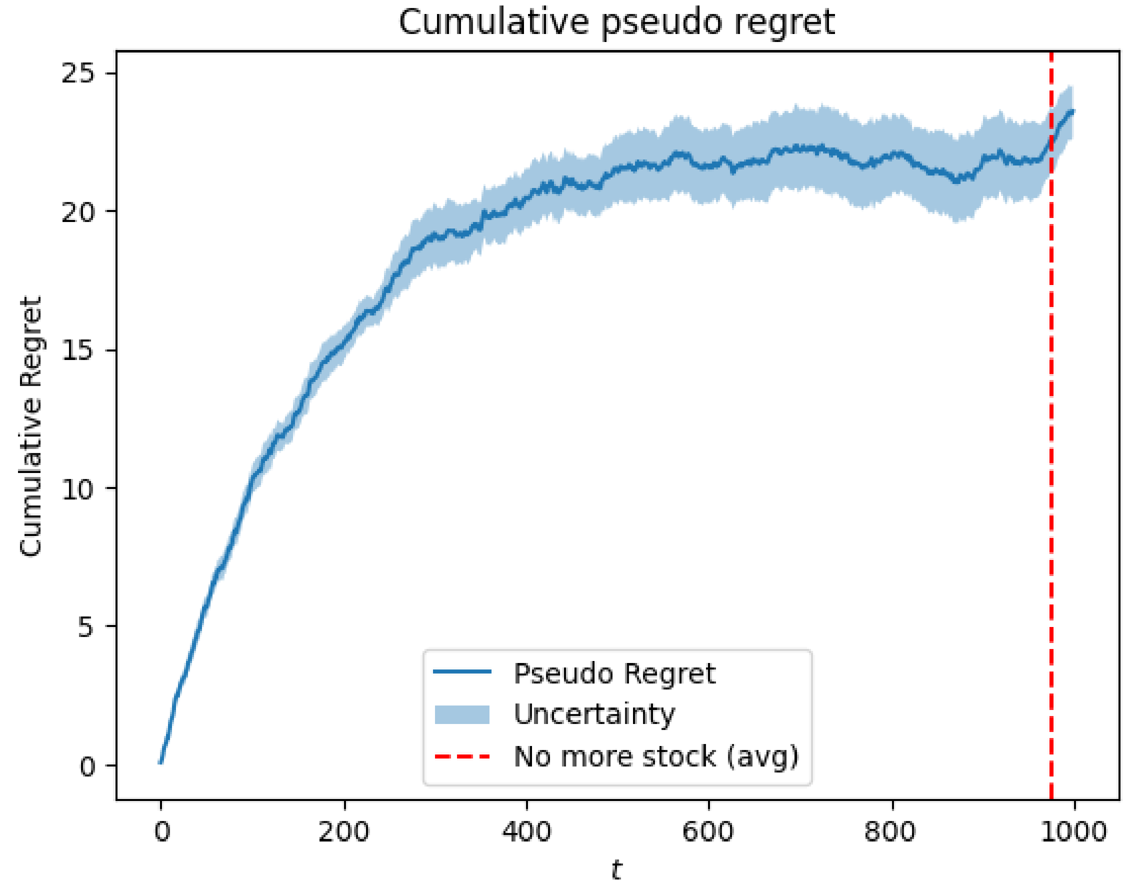
$T = 1000$ $B = 400$ $P = \{0.1, 0.2, \dots, 1, 1.1\}$
$c = 0.1$ $n_{trails} = 10$

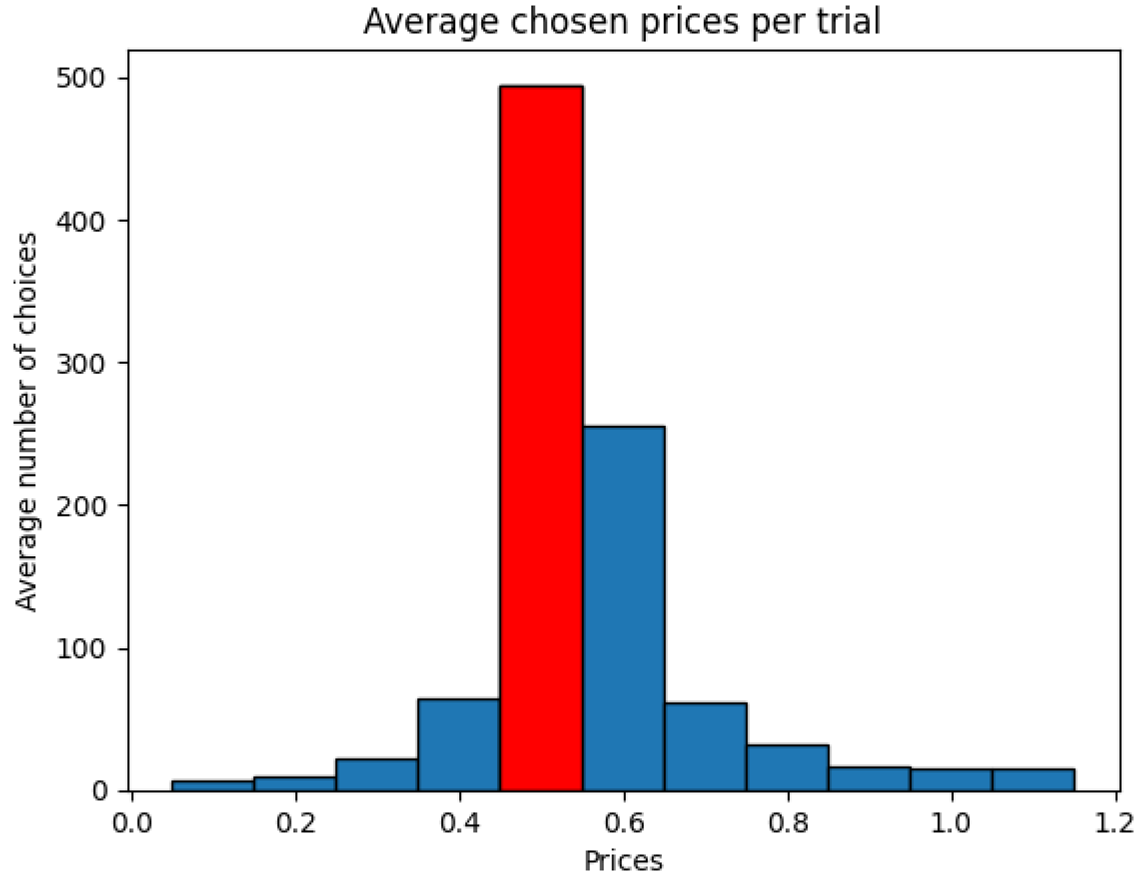




FRAMEWORK: SETTING 2

$T = 1000$ $B = 500$ $P = \{0.1, 0.2, \dots, 1, 1.1\}$
$c = 0.1$ $n_{trails} = 10$ $R = 3$





REQUIREMENT 4

TASK

Design a primal-dual algorithm to simulate sellings of multiple products with the inventory constraint

Setting 1:

- Stochastic environment
- $N > 1$ and their production costs $\boldsymbol{c} = (c_i)^N{}_{i=1}$
- $\boldsymbol{P}$ set of superarms

Setting 2:

- Highly non-stationary environment
- $N > 1$ and their production costs $\boldsymbol{c} = (c_i)^N{}_{i=1}$
- $\boldsymbol{P}$ set of superarms

REQUIREMENT 4

ALGORITHM 4

INPUTS: T, P, N, B

INITIALIZATION: $\lambda = (1, \ldots, 1),\ \rho = \frac{B}{T},\ \eta = T^{-\frac{1}{2}}$

For $t = 1,2, \ldots T$:

- Sample $\boldsymbol{p}_t$ using N regret minimizers (**Hedge**)
- Observe $F_t(\boldsymbol{p}_t)$ and $S_t(\boldsymbol{p}_t)$ $\forall\, \boldsymbol{p} \in \boldsymbol{P}$ (Full feedback)

  For $n = 1 \ldots N$:

  - Compute: $L_t^{(n)}(p) = f_t^{(n)}(p) - \lambda_t^{(n)}(p)\left(s_t^{(n)}(p) - \frac{\rho}{N}\right)\ \forall p \in P$
  - $\lambda_{t+1}^{(n)}(p) = \Pi_{\left[0,\frac{1}{\rho}\right]}\left(\lambda_t^{(n)}(p) - \eta\left(\frac{\rho}{N} - s_t^{(n)}(p)\right)\right) \forall p \in P$
  - Update Hedge passing $l_t^{(n)}(p) = 1 - L_{t,norm}^{(n)}(p)\ \forall p \in P$

- $B = B - S_t(\boldsymbol{p}_t) \rightarrow$ stop if $B < 1$

NORMALIZATION OF THE LAGRANGIAN

$$L_{norm} = min\left(1, max\left(0, \frac{L(p) - L_{min}}{L_{MAX} - L_{min}}\right)\right)$$

Where:

- $L_{MAX} = (max_{p \in P \setminus \{\underset{P}{max}(p)\}}(p) - c) + max\big(\lambda\rho, \lambda(1-\rho)\big)$
- $L_{min} = min(\lambda\rho, \lambda(1-\rho))$

REQUIREMENT 4

IMPLEMENTATION: SETTING 1

- Valuations are $\mathcal{U}$ – distributed with a different mean for each product (**independece assumption**)
- Definition of **Buyer, Hedge Agent** and **MultiplicativePacingSeller**
- **Clairvoyant** (same as Requirement 2):
  - Computation of the best distribution solving $LP(F^*(\boldsymbol{\gamma}), S^*(\boldsymbol{\gamma}), \boldsymbol{P})$ problem
  - Expected utility

IMPLEMENTATION: SETTING 2

- Generation of valuations by partitioning $[T]$ in blocks:
  - Each block consits of R rounds
  - In odd blocks: $\mathcal{U}$- distributed, with a changing mean
  - In even blocks: $Beta(\alpha, \alpha)$-distributed with a changing α
  - Generated using ***beta*** from ***scipy.stats***
- Definition of **Buyer**, **Hedge Agent** and **MultiplicativePacingSeller**
- **Clairvoyant**:

$$F^* = \sum_{\boldsymbol{p}} \gamma(\boldsymbol{p}) \sum_{i=1}^{N} (p^{(i)} - c^{(i)}) Pr_{emp}(p^{(i)} < v^{(i)})$$

Where:

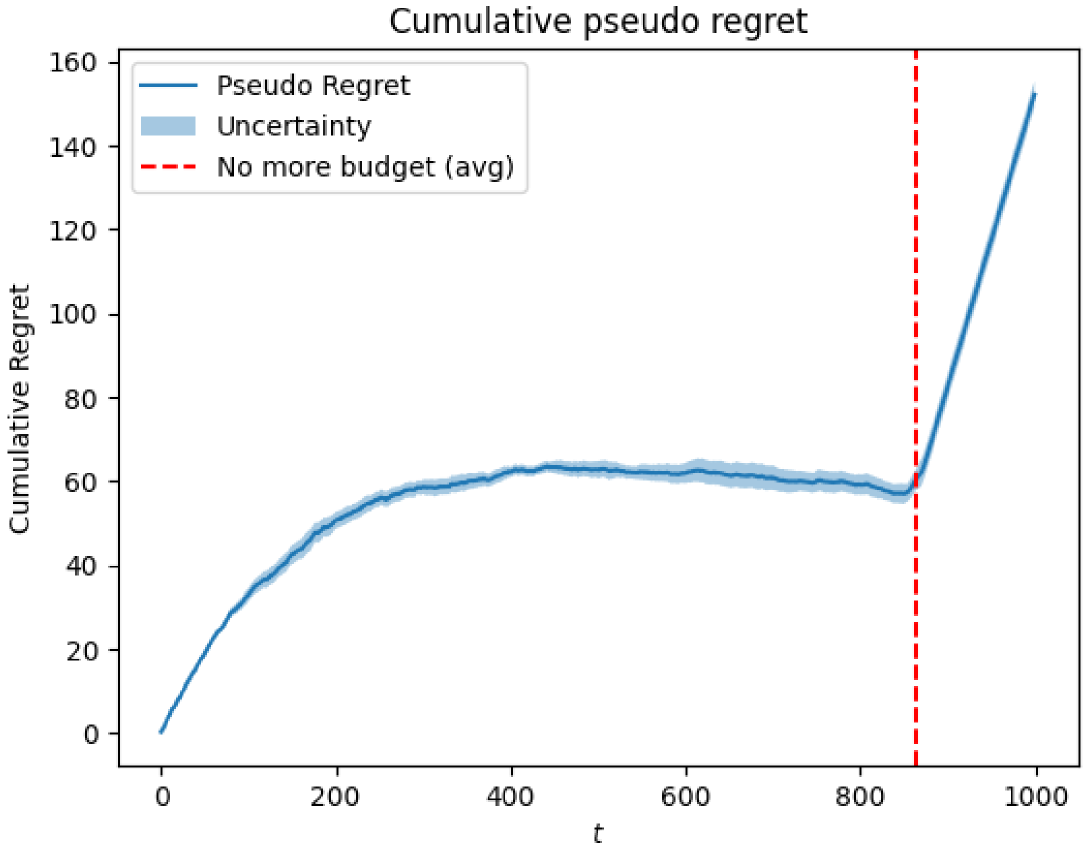
$$Pr_{emp} = \text{ empirical probability} = \frac{\#\, of \text{ successes}}{T}$$

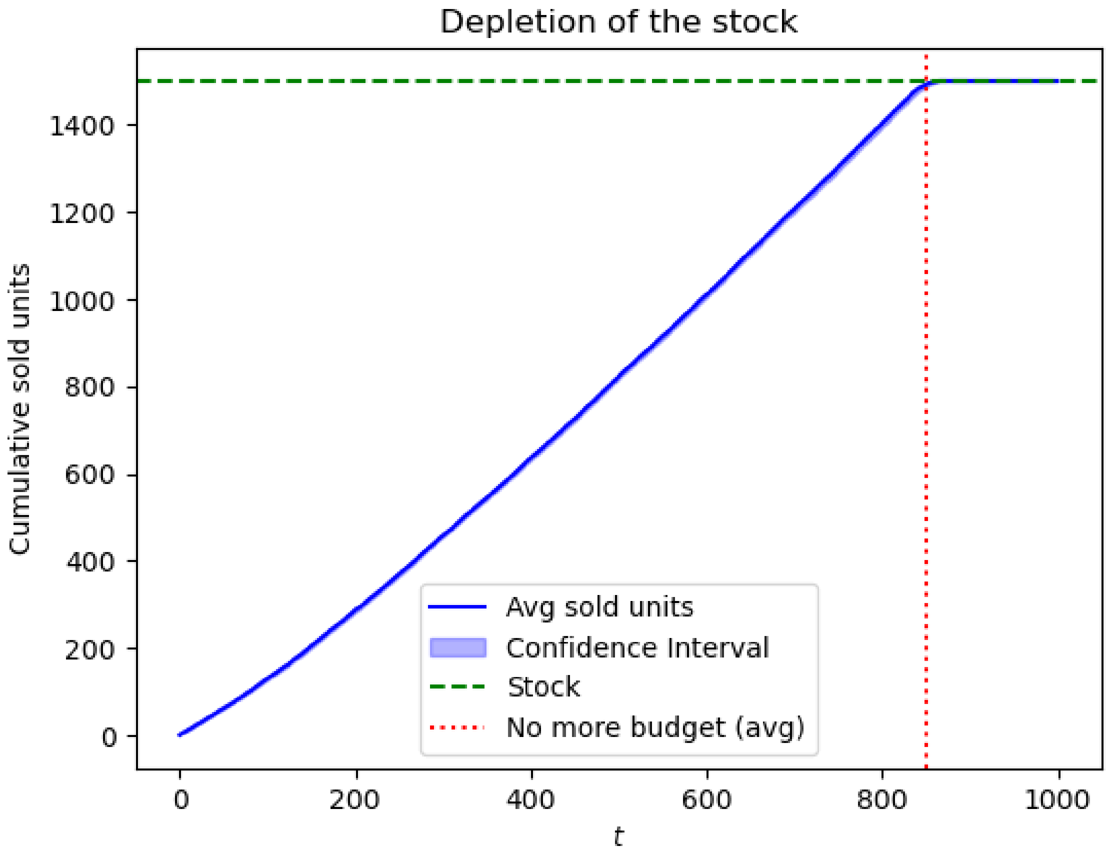
$\boldsymbol{\gamma}(\boldsymbol{p})$ founded by $LP(F^*, S^*, \boldsymbol{P})$

REQUIREMENT 4
Results

FRAMEWORK: SETTING 1

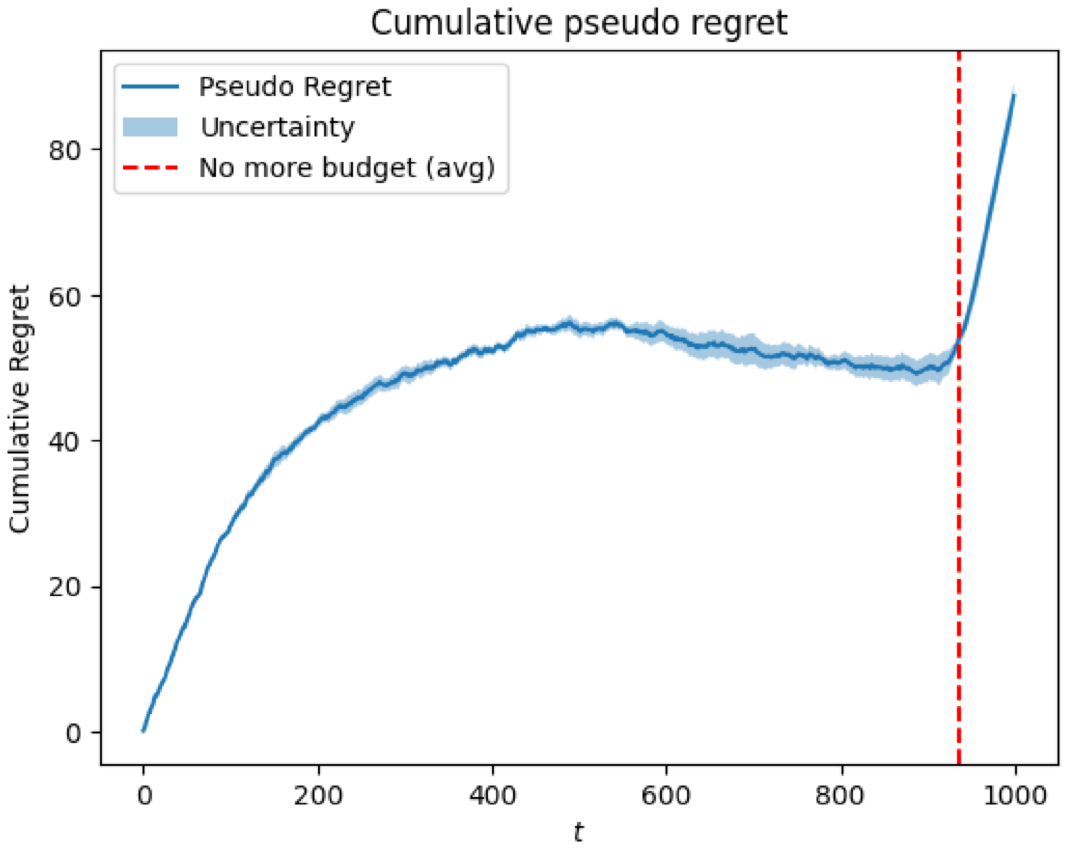
$$N = 3 \qquad T = 1000$$
$$B = 1500 \qquad P = \{0.1, 0.2, \dots, 1, 1.1\}$$
$$c_i = 0.1\ \forall i \qquad n_{trails} = 5$$

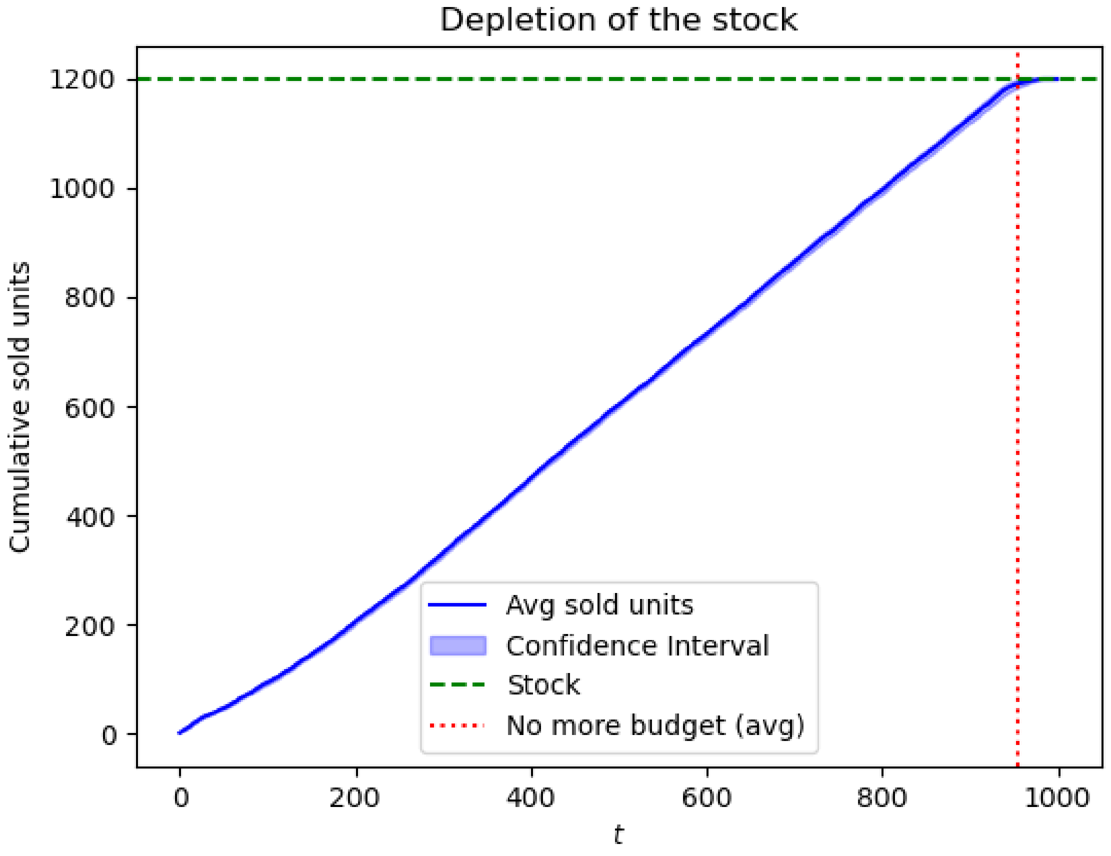




FRAMEWORK: SETTING 2

$$N = 3 \qquad T = 1000$$
$$B = 1200 \qquad P = \{0.1, 0.2, \dots, 1, 1.1\}$$
$$c_i = 0.1\ \ \forall i \qquad n_{trails} = 5 \qquad R = 3$$





REQUIREMENT 5

Setting:

- Slightly non-stationary environment
- $N > 1$ and their production costs $\boldsymbol{c} = (c_i)^N{}_{i=1}$
- $\boldsymbol{P}$ set of superarms

Generation of valuations by partitioning $[T]$ in blocks:

- Each block consits of R rounds
- $\mathcal{U}$ – distributed , with a changing mean
- Means change between blocks

TASK

Extend Combinatorial-UCB with sliding window

REQUIREMENT 5

ALGORITHM 5

INPUTS: $T, P, N, B, W, \rho = \frac{B}{T}$

For $t = 1,2, \dots T$:

- Compute $F_{t,W}^{UCB}(\boldsymbol{p}) = \sum_{i=1}^{N} f_{t,W}^{UCB}(p^{(i)}) \;\; \forall \boldsymbol{p} \in \boldsymbol{P}$
- Compute $S_{t,W}^{LCB}(\boldsymbol{p}) = \sum_{i=1}^{N} s_{W}^{LCB}(p^{(i)}) \;\; \forall \boldsymbol{p} \in \boldsymbol{P}$
- Retrieve $\boldsymbol{\gamma}_t$ solving $LP(F_{t,W}^{UCB}, S_{t,W}^{LCB}, \boldsymbol{P})$ and sample $\boldsymbol{p}_t \sim \boldsymbol{\gamma}_t$
- Observe $S_t(\boldsymbol{p}_t)$ and $F_t(\boldsymbol{p}_t)$
- Decrease the inventory: $B = B - S_t(\boldsymbol{p}_t)$
- $B = B - s_t(p_t) \rightarrow$ stop if $B < N$

IMPLEMENTATION

- Definition of **Buyer** and **SW-UCBLikeSeller**
- **Sliding Window: cache matrices** of dim (N, W, K)
  - Store $f_{t,W}(p^{(i)}), s_{t,W}(p^{(i)}) \; \forall (i, p) \in \{1 \dots N\} \text{x} P$
  - Remove the oldest observations by shifting the rows
- **Clairvoyant:** it gains the best fixed expected utility per round:

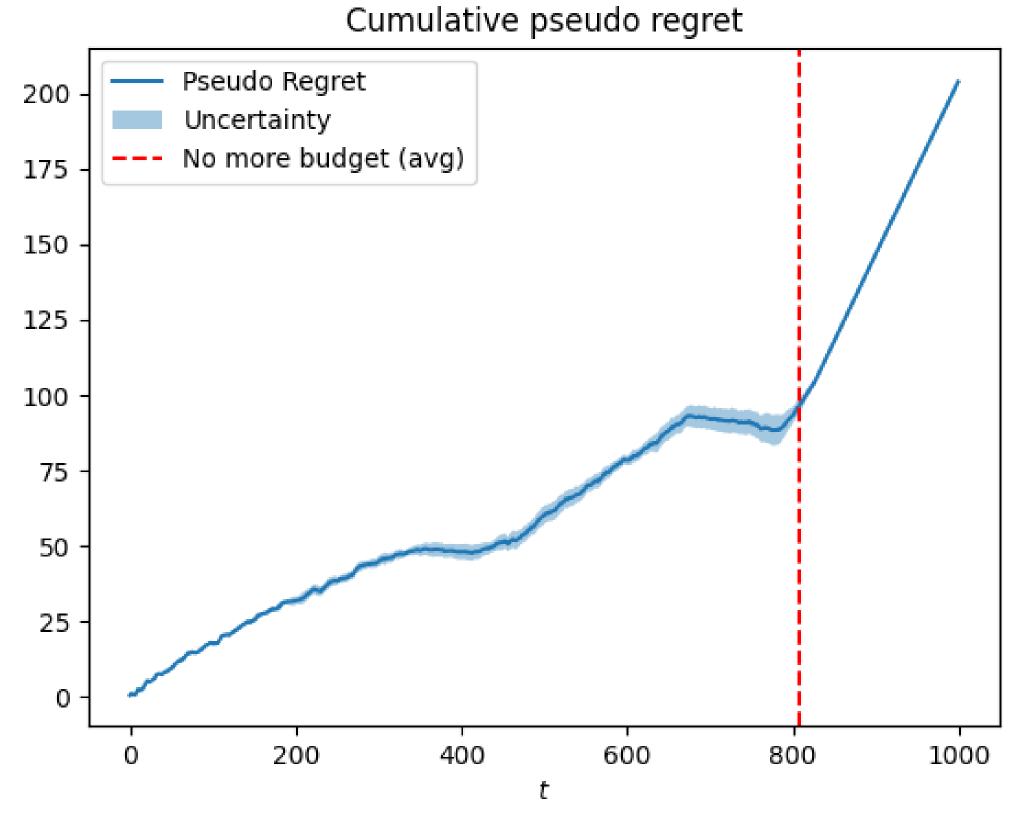
$$\langle F \rangle = \frac{1}{T} \sum_{j=1}^{\#\, of\, blocks} F_j^*(\boldsymbol{\gamma}) \cdot R_j$$

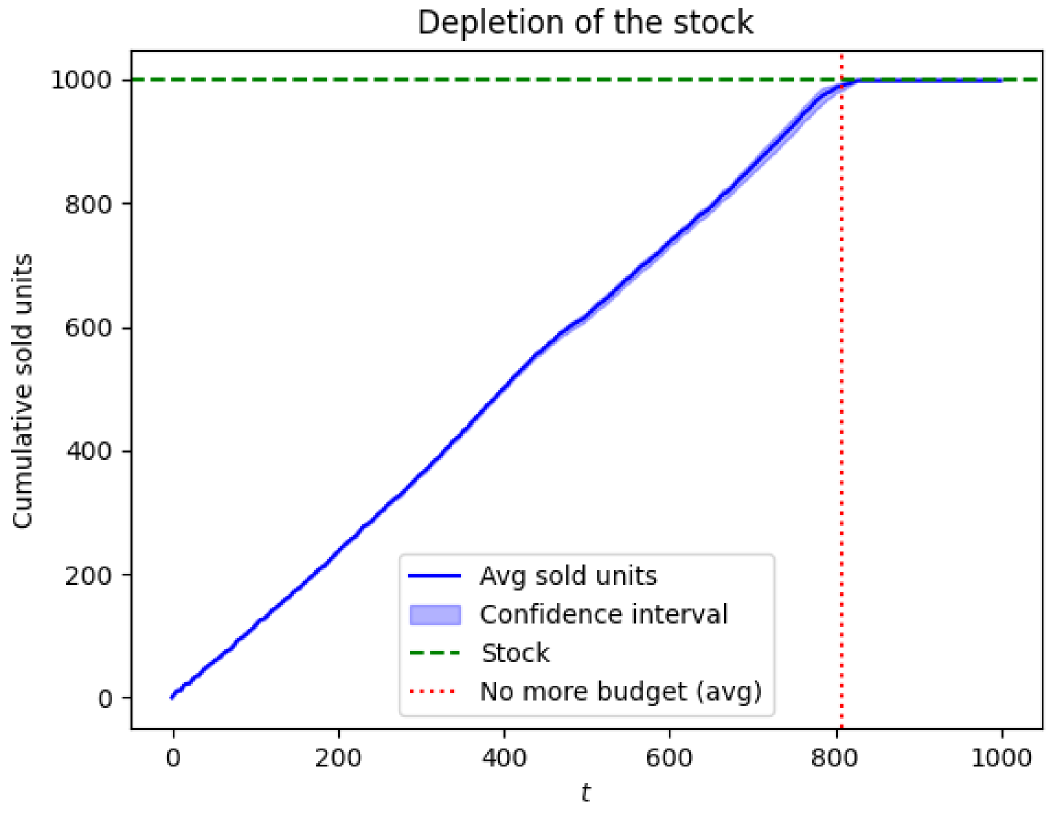
  - R_j = # of rounds in the block number j
  - $F_j^*(\boldsymbol{\gamma}) = \sum_{\boldsymbol{p}} \gamma(\boldsymbol{p}) \sum_{i=1}^{N} (p^{(i)} - c^{(i)}) Pr_{\boldsymbol{j}}(p^{(i)} < v^{(i)})$
  - $\boldsymbol{\gamma}(\boldsymbol{p})$ founded by $LP(F_j^*(\boldsymbol{\gamma}), S_j^*(\boldsymbol{\gamma}), \boldsymbol{P})$

REQUIREMENT 5
Results

FRAMEWORK 1

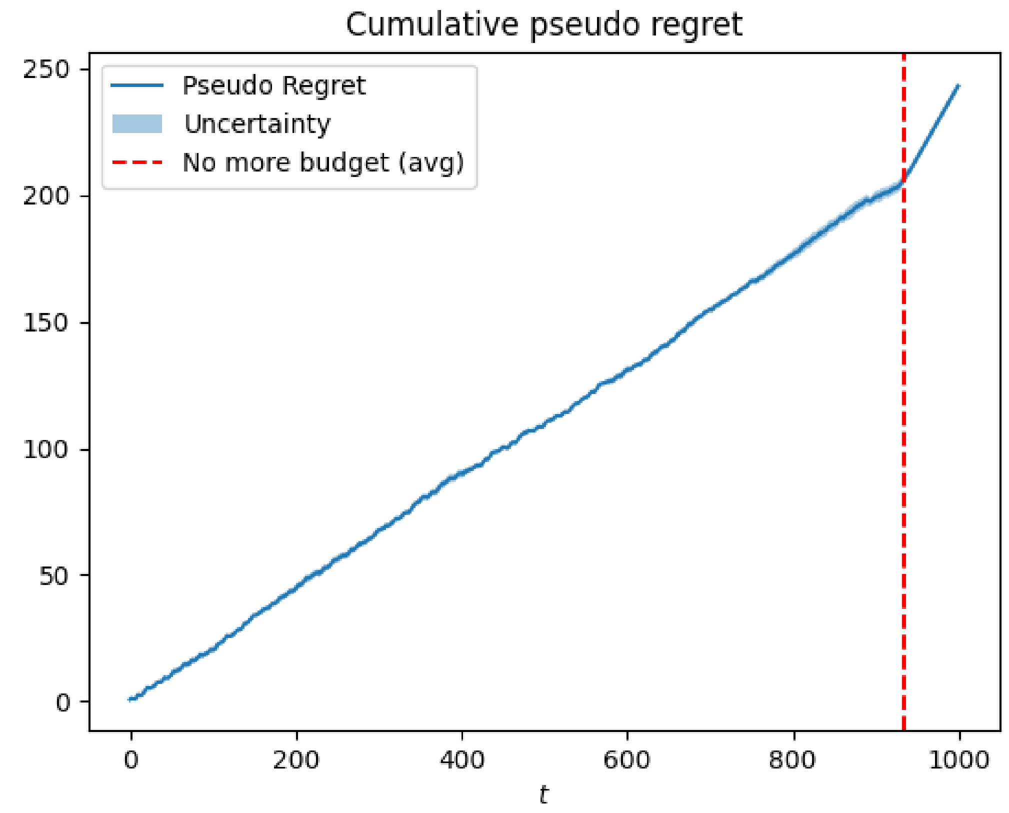
$$N = 3 \quad T = 1000$$
$$B = 1000 \quad P = \{0.1, 0.2, \dots, 1, 1.1\}$$
$$R = \frac{T}{3} \quad W = R$$
$$c_i = 0.1 \ \forall i \qquad n_{trails} = 3$$

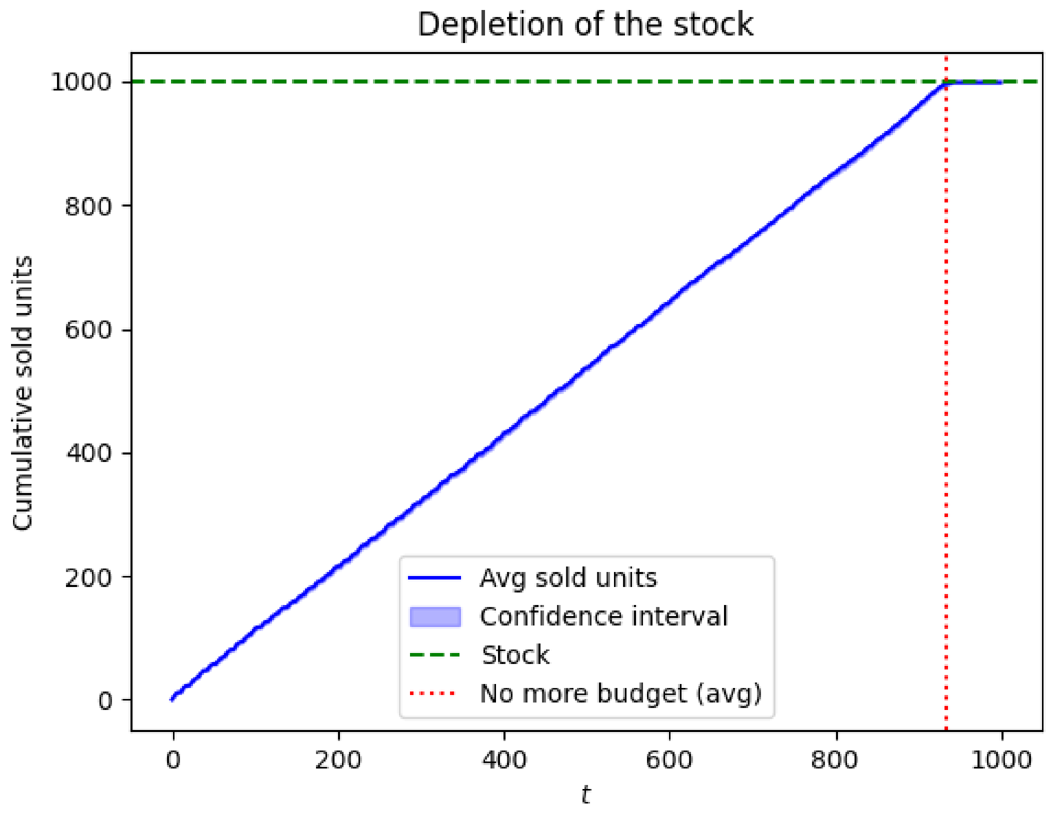




FRAMEWORK 2

$$N = 3 \quad T = 1000$$
$$B = 1000 \quad P = \{0.1, 0.2, \dots, 1, 1.1\}$$
$$R = \frac{T}{3} \quad W = \sqrt{T}$$
$$c_i = 0.1 \ \forall i \qquad n_{trails} = 3$$





COMPARISON

FRAMEORK

$$N = 3 \quad T = 1000 \quad B = 750 \quad P = \{0.1, 0.2, \dots, 1, 1.1\}$$
$$R = \frac{T}{3} \quad W = R \quad c_i = 0.1\ \forall i \quad n_{trails} = 3$$

ALGORITHM 4: RESULTS

- Average cumulative reward: 272,50
- Average depletion time: 837

ALGORITHM 5: RESULTS

- Average cumulative reward: 268,43
- Average depletion time: 605