Decentralized Multi-Agent Active Search for Sparse Signals: Supplementary Material

Ramina Ghods[1] **Arundhati Banerjee**[1] **Jeff Schneider**[1]

[1]School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA

ABSTRACT

This appendix includes derivations for the proposed Thompson Sampling algorithms Laplace-TS, SPATS and LATSI, discussion and example on failure mode of Laplace-TS, theoretical proofs for Theorem 1 and Theorem 2 and additional numerical results.

Notation Lowercase and uppercase boldface letters represent column vectors and matrices, respectively. For a matrix $\mathbf{A}$, its transpose $\mathbf{A}^{\mathrm{T}}$, and the kth row and ℓth column entry is $A_{k,\ell}$. For a vector $\mathbf{a}$, the kth entry is a_k, and the sub-vector containing the ith to jth entries (excluding j) is $\mathbf{a}[i:j]$ (Python notation). The ℓ_1 and ℓ_2-norm of $\mathbf{a}$ are denoted by $\|\mathbf{a}\|_1$ and $\|\mathbf{a}\|_2$, and $|\mathbf{a}|$ represents its absolute value applied element-wise. The Kronecker product is $\otimes$, and the trace operator is $\mathrm{tr}(\cdot)$. The $N \times N$ identity matrix is denoted by $\mathbf{I}_N$. $\mathrm{diag}(\mathbf{a})$ is a square matrix with $\mathbf{a}$ on the main diagonal. $\emptyset$ denotes an empty set and for a set $\mathcal{S}$, $|\mathcal{S}|$ shows the number of elements in that set. $\mathcal{A} \wedge \mathcal{B}$ depicts the logical *AND* between two sets $\mathcal{A}$ and $\mathcal{B}$. We use symbols $\triangleq$ and $\mathbb{1}(.)$ as the symbols for mathematical definition and indicator function, respectively.

1 LAPLACE-TS: MULTI-AGENT THOMPSON SAMPLING WITH SPARSE PRIOR

We will now derive the posterior sampling and design stages of Laplace-TS algorithm introduced in Section 3.2. As detailed in this section, we use a Laplace sparse prior $p(\boldsymbol{\beta}) = \frac{1}{(2b)^n} \exp(-\frac{\|\boldsymbol{\beta}\|_1}{b})$ and likelihood distribution $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) = \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2 \mathbf{I}_{t-1})$ for this problem.

Posterior Sampling In this stage (also referred to as inference stage in active learning algorithms), each agent j regularly updates its posterior distribution of $\boldsymbol{\beta}$ given the data set $\mathbf{D}_t^j$ available to it. Using Bayes rule we have:

$$p(\boldsymbol{\beta}|\mathbf{D}_t^j) = p(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y}) = \frac{1}{Z} p(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) p(\boldsymbol{\beta}).$$

Unfortunately, computing the normalization factor Z above is intractable for a Bayesian likelihood distribution with a Laplace prior. To compute this posterior, we borrow ideas from Figueiredo [2003]. In particular, we substitute the Laplace prior distribution with a zero-mean Gaussian prior per entry $p(\beta_i|\tau_i) = \mathcal{N}(0, \tau_i)$, where its variance τ_i has an exponential hyper prior $p(\tau_i) = \frac{\eta}{2} \exp(-\frac{\eta}{2}\tau_i)$ with $\tau_i \geq 0$.

Integrating out τ_i to compute $p(\beta_i)$, we see that this new prior is equivalent to the original Laplace prior per entry β_i:

$$p(\beta_i) = \int p(\beta_i|\tau_i) p(\tau_i) \, \mathrm{d}\tau_i = \frac{\sqrt{\eta}}{2} \exp(-\sqrt{\eta}|\beta_i|).$$

Here, $\sqrt{\eta} = 1/b$ is the scaling hyperparameter in Laplace distribution and for best performance needs to be tuned given the sparsity rate of the original signal $\boldsymbol{\beta}$. If we were able to observe variance τ_i, the posterior distribution of $\boldsymbol{\beta}$ given data set $\mathbf{D}_t^j$ would become:

$$p(\boldsymbol{\beta}|\mathbf{D}_t^j, \boldsymbol{\tau}) = \frac{1}{Z} \, p(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) p(\boldsymbol{\beta}|\boldsymbol{\tau})$$

Supplement for the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI 2021).

$$= \frac{1}{Z}\,\mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}_{t-1}) \times \mathcal{N}(0, \text{diag}(\boldsymbol{\tau}))$$
$$= \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\beta}}(\boldsymbol{\tau}), \boldsymbol{\Sigma}_{\boldsymbol{\beta}}(\boldsymbol{\tau})), \tag{8}$$

with,

$$\boldsymbol{\Sigma}_{\boldsymbol{\beta}}(\boldsymbol{\tau}) = \left((\text{diag}(\boldsymbol{\tau}))^{-1} + \frac{1}{\sigma^2}\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1},$$
$$\boldsymbol{\mu}_{\boldsymbol{\beta}}(\boldsymbol{\tau}) = \frac{1}{\sigma^2}\boldsymbol{\Sigma}_{\boldsymbol{\beta}}(\boldsymbol{\tau})\mathbf{X}^{\mathrm{T}}\mathbf{y}. \tag{9}$$

Here, $\boldsymbol{\tau}$ is the vector containing all elements of τ_i for $i = 1, ..., n$. Since we cannot observe $\boldsymbol{\tau}$, we use Gibbs sampling [Gelfand and Smith, 1990] to iteratively sample $\boldsymbol{\tau}$ and $\boldsymbol{\beta}$ from their conditional posterior distributions. For the given prior, Bae and Mallick [2004] has computed the conditional distributions to iteratively sample $\boldsymbol{\tau}$ and $\boldsymbol{\beta}$ as follows:

$$\boldsymbol{\beta}^{\star} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\beta}}(\tilde{\boldsymbol{\tau}}), \boldsymbol{\Sigma}_{\boldsymbol{\beta}}(\tilde{\boldsymbol{\tau}}))$$
$$\tilde{\tau}_i^{-1} \sim InvGauss(\frac{\sqrt{\eta}}{\beta_i^{\star}}, \eta), \quad i = 1, ..., n. \tag{10}$$

Design In this stage, agent j computes the expected reward $\lambda^{+}(\boldsymbol{\beta}^{\star}, \mathbf{D}_t^j, \mathbf{x})$ and optimize it for best sensing action $\mathbf{x}_t$. Let us use the reward function $\lambda(\boldsymbol{\beta}^{\star}, \mathbf{D}_t^j \cup (\mathbf{x}, y)) = -\|\boldsymbol{\beta}^{\star} - \hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{x}, y))\|_2^2$. In order to compute the expected reward in (3), we need to design an estimator $\hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{x}, y))$ whose expectation we can compute. While there has been many sparse recovery algorithms proposed in the literature [Marques et al., 2018], for many of these well-known thresholding or iterative algorithms (e.g. [Pati et al., 1993, Needell and Tropp, 2010, Blumensath and Davies, 2009, Daubechies et al., 2004, Maleki, 2011]) computing the expectation in (3) is not tractable. Alternatively, we propose using maximum a posteriori (MAP) estimate of the posterior $p(\boldsymbol{\beta}|\mathbf{D}_t^j, \boldsymbol{\tau})$ in (8) where we estimate $\boldsymbol{\tau}$ using Expectation-Maximization (EM) proposed by Figueiredo [2003] with the E-step and M-step for $p = 1, ..., P$ iterations as follows:

$$\textbf{E-step}: \boldsymbol{\tau}^{(p)} = |\hat{\boldsymbol{\beta}}^{(p-1)}|/\eta$$
$$\textbf{M-step}: \hat{\boldsymbol{\beta}}^{(p)} = \boldsymbol{\mu}_{\boldsymbol{\beta}}(\boldsymbol{\tau}^{(p)}). \tag{11}$$

Since many elements of $\hat{\boldsymbol{\beta}}^{(p)}$ and consequently $\boldsymbol{\tau}^{(p)}$ are expected to be zero, to avoid inverting $\text{diag}(\boldsymbol{\tau})$ we can rewrite the M-step as follows: $\hat{\boldsymbol{\beta}}^{(p)} = \frac{1}{\sigma^2}\boldsymbol{\Sigma}^{(p)}\mathbf{X}^{\mathrm{T}}\mathbf{y}$, with $\boldsymbol{\Sigma}^{(p)} = \mathbf{U}\left(\mathbf{I}_n + \frac{1}{\sigma^2}\mathbf{U}\mathbf{X}^{\mathrm{T}}\mathbf{X}\mathbf{U}\right)^{-1}\mathbf{U}$ and $\mathbf{U} = \left(\text{diag}(\boldsymbol{\tau}^{(p)})\right)^{1/2}$.

Since the posterior is Gaussian, its MAP estimate is the mean of the distribution, i.e. $\hat{\boldsymbol{\beta}}^{(P)}$ in (11). Adding $(\mathbf{x}, y)$ to the data set $\mathbf{D}_t^j$, our MAP estimate becomes:

$$\hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{x}, y)) = \underbrace{\mathbf{U}\left(\sigma^2\mathbf{I}_n + \mathbf{U}\begin{bmatrix}\mathbf{X}^{\mathrm{T}} & \mathbf{x}\end{bmatrix}\begin{bmatrix}\mathbf{X} \\ \mathbf{x}^{\mathrm{T}}\end{bmatrix}\mathbf{U}\right)^{-1}\mathbf{U}}_{\triangleq \mathbf{Q}}\begin{bmatrix}\mathbf{X}^{\mathrm{T}} & \mathbf{x}\end{bmatrix}\begin{bmatrix}\mathbf{y} \\ y\end{bmatrix}.$$

Now, we can derive the expected reward in (3) as follows:

$$\begin{aligned}\lambda^{+}(\boldsymbol{\beta}^{\star}, \mathbf{D}_t^j, \mathbf{x}) &= \mathbb{E}_{y|\mathbf{x},\boldsymbol{\beta}^{\star}}\left[-\|\boldsymbol{\beta}^{\star} - \hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{x}, y))\|_2^2\right] = -\,\mathbb{E}_{y|\mathbf{x},\boldsymbol{\beta}^{\star}}\left[\|\mathbf{Q}\mathbf{X}^{\mathrm{T}}\mathbf{y} + \mathbf{Q}\mathbf{x}y - \boldsymbol{\beta}^{\star}\|_2^2\right] \\ &= -\|\mathbf{Q}\mathbf{X}^{\mathrm{T}}\mathbf{y} - \boldsymbol{\beta}^{\star}\|_2^2 - \|\mathbf{Q}\mathbf{x}\|_2^2\,\mathbb{E}_{y|\mathbf{x},\boldsymbol{\beta}^{\star}}\left[y^2\right] - 2\left(\mathbf{Q}\mathbf{X}^{\mathrm{T}}\mathbf{y} - \boldsymbol{\beta}^{\star}\right)^{\mathrm{T}}\mathbf{Q}\mathbf{x}\,\mathbb{E}_{y|\mathbf{x},\boldsymbol{\beta}^{\star}}[y] \\ &= -\|\mathbf{Q}\mathbf{X}^{\mathrm{T}}\mathbf{y} - \boldsymbol{\beta}^{\star}\|_2^2 - \|\mathbf{Q}\mathbf{x}\|_2^2\left(\sigma^2 + (\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta}^{\star})^2\right) - 2\left(\mathbf{Q}\mathbf{X}^{\mathrm{T}}\mathbf{y} - \boldsymbol{\beta}^{\star}\right)^{\mathrm{T}}\mathbf{Q}\mathbf{x}\mathbf{x}^{\mathrm{T}}\boldsymbol{\beta}^{\star}.\end{aligned} \tag{12}$$

With the posterior sampling and design stage developed, we can now run Algorithm 1 for the active search problem in Section 2. Let us call this algorithm Laplace-TS. In the next section, we will evaluate the performance of this algorithm.

2 FAILING PERFORMANCE OF LAPLACE-TS IN SINGLE AGENT SETTING

Figure 7 illustrates simulation results of the full recovery performance of Laplace-TS compared to a simple point algorithm that exhaustively searches the entire action space one location at a time with one agent. Here, we are plotting full recovery rate of vector $\boldsymbol{\beta}$ with size $n = 128$ as a function of number of measurements T for two sparsity rates of $k = 1$ and $k = 5$. The curves show mean and standard error of full recovery over 50 random trials. This figure demonstrates that, unfortunately,

Laplace-TS with one agent leads to poor performance that is on par with a point-wise algorithm that exhaustively searches all locations one at a time.

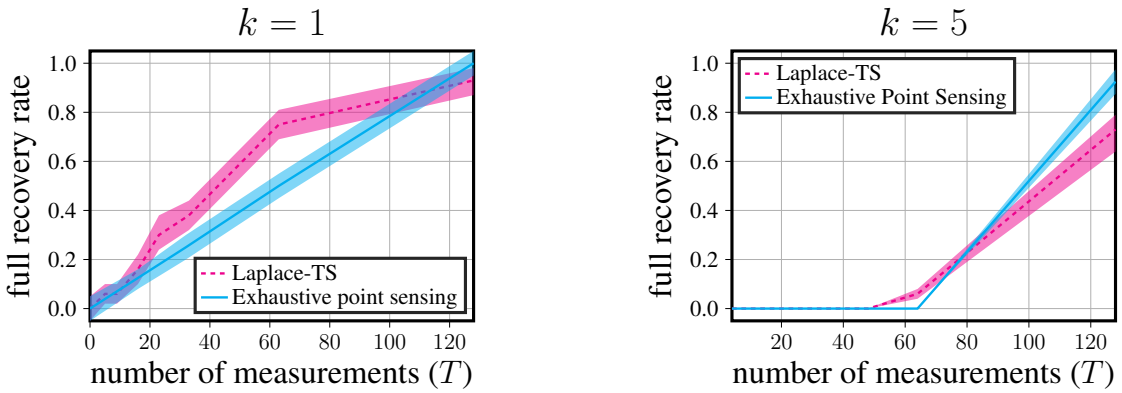


Figure 7: Full recovery rate of Laplace-TS with single agent for sparsity rate $k = 1$

We can associate this poor performance with one of the failure modes of TS discussed in Sec. 8.2 of the tutorial by Russo et al. [2018]. According to this tutorial, TS faces a dilemma when solving certain kinds of active learning problems. One such scenario are problems that require a careful assessment of information gain. In general, by optimizing the expected reward, TS always restricts its actions to those that have a chance in being optimal. However, in certain active learning problems, suboptimal actions can carry additional information regarding the parameter of interest.

We provide the following example to better understand the situation in our problem formulation. Let us assume there is no noise in the sensing model (1) ($\epsilon_t = 0$) and that there is only one non-zero element ($k = 1$). Under such conditions, the problem amounts to finding the location of the non-zero element in $\boldsymbol{\beta}$ which we call $\tilde{i}$. Thus, for every action $\mathbf{x}_t$, the observation $y_t = \mathbf{x}_t^{\mathrm{T}}\boldsymbol{\beta}$ is non-zero if $\tilde{i}$ is in the support of $\mathbf{x}_t$ and is zero, otherwise. Clearly, a binary search can locate $\tilde{i}$ in $\log(n)$ steps. TS, however, in each time step estimates the sample $\boldsymbol{\beta}^\star$ with $\hat{\boldsymbol{\beta}}$ and picks the sensing action $\mathbf{x}_t = \hat{\boldsymbol{\beta}}$. Or, in case of perfect estimation, we will have $\mathbf{x}_t = \hat{\boldsymbol{\beta}} = \boldsymbol{\beta}^\star$. So, unless $\boldsymbol{\beta}^\star$ is the true $\boldsymbol{\beta}$, observation $\mathbf{y}_t$ would be zero and TS will only manage to eliminate support of $\boldsymbol{\beta}^\star$ from the list of possibly correct supports. Therefore, TS ends up eliminating wrong supports one location at a time which explains its on par performance to an exhaustive point sensor. Similarly, for $k > 1$ TS is always limited to picking sensing actions $\mathbf{x}_t$ that are in the support of the sparse estimates of the samples. Adding the region sensing constraint will only aggravate this problem by further shrinking this support set. In the next section, we will modify Laplace-TS and propose two algorithms that can bypass this failure mode.

3 SPATS: THOMPSON SAMPLING WITH BLOCK SPARSE PRIOR

In what follows, we will derive the posterior sampling and design stages for SPATS algorithm proposed in Section 4.1. SPATS is an asynchronous multi-agent TS algorithm using a block sparse prior for the problem formulation in Section 2.

As discussed in Section 4.1, we use the following prior and likelihood distributions. The likelihood function is $p(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) = \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}_{t-1})$ from (2). For prior distribution, as discussed in Section 4.1 we use a block sparse prior $p(\boldsymbol{\beta}) = \mathcal{N}(\mathbf{0}_{n\times 1}, \boldsymbol{\Sigma}_0)$, where:

$$\boldsymbol{\Sigma}_0 = \mathrm{diag}(\boldsymbol{\gamma}) \otimes \mathbf{B}, \tag{13}$$

with $\boldsymbol{\gamma} \in \mathbb{R}^M$ and $\mathbf{B} \in \mathbb{R}^{L\times L}$ ($M = n/L$) as hyperparameters.

Posterior Sampling Similar to (8) and (9) in computing the posterior distribution for agent j, the posterior with a Gaussian likelihood and Gaussian prior becomes:

$$p(\boldsymbol{\beta}|\mathbf{D}_t^j, \boldsymbol{\gamma}, \mathbf{B}) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\beta}}(\boldsymbol{\gamma}, \mathbf{B}), \boldsymbol{\Sigma}_{\boldsymbol{\beta}}(\boldsymbol{\gamma}, \mathbf{B})), \tag{14}$$

with,

$$\boldsymbol{\Sigma}_{\boldsymbol{\beta}}(\boldsymbol{\gamma}, \mathbf{B}) = ((\boldsymbol{\Sigma}_0)^{-1} + \frac{1}{\sigma^2}\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1},$$

$$\boldsymbol{\mu}_{\boldsymbol{\beta}}(\boldsymbol{\gamma}, \mathbf{B}) = \frac{1}{\sigma^2}\boldsymbol{\Sigma}_{\boldsymbol{\beta}}(\boldsymbol{\gamma}, \mathbf{B})\mathbf{X}^{\mathrm{T}}\mathbf{y}.$$

Using Expectation-Maximization proposed by Zhang and Rao [2011], we can estimate the hyperparameters $\boldsymbol{\gamma}$ and $\mathbf{B}$ for $p = 1, ..., P$ iterations as follows.

$$\begin{aligned}
\textbf{E-step}: \quad \boldsymbol{\mu}_{\boldsymbol{\beta}}^{(p)} &= \boldsymbol{\mu}_{\boldsymbol{\beta}}(\boldsymbol{\gamma}^{(p-1)}, \mathbf{B}^{(p-1)}), \\
\boldsymbol{\Sigma}_{\boldsymbol{\beta}}^{(p)} &= \boldsymbol{\Sigma}_{\boldsymbol{\beta}}(\boldsymbol{\gamma}^{(p-1)}, \mathbf{B}^{(p-1)}) \\
\textbf{M-step}: \quad \gamma_m^{(p)} &= \frac{\mathrm{tr}(\mathbf{B}^{-1}(\boldsymbol{\Sigma}_{\boldsymbol{\beta}}^m + \boldsymbol{\mu}_{\boldsymbol{\beta}}^m \left(\boldsymbol{\mu}_{\boldsymbol{\beta}}^m\right)^{\mathrm{T}}))}{L}, \quad (m = 1, ..., M) \\
\mathbf{B}^{(p)} &= \frac{1}{M}\sum_{m=1}^{M} \frac{\Sigma_{\boldsymbol{\beta}}^m + \boldsymbol{\mu}_{\boldsymbol{\beta}}^m \left(\boldsymbol{\mu}_{\boldsymbol{\beta}}^m\right)^{\mathrm{T}}}{\gamma_m},
\end{aligned} \tag{15}$$

with $\boldsymbol{\mu}_{\boldsymbol{\beta}}^m = \mu_{\boldsymbol{\beta}}^{(p)}[(m-1) \times L : m \times L]$, and $\boldsymbol{\Sigma}_{\boldsymbol{\beta}}^m = \boldsymbol{\Sigma}_{\boldsymbol{\beta}}^{(p)}[(m-1) \times L : m \times L, (m-1) \times L : m \times L]$.

Design Next, for the estimator of agent j, $\hat{\boldsymbol{\beta}}(\mathbf{D}_t^j)$, we use the MAP estimator given by $\boldsymbol{\mu}_{\boldsymbol{\beta}}^{(P)}$. Particularly, we have:

$$\hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{x}, y)) = \underbrace{\left(\sigma^2\boldsymbol{\Sigma}_0^{-1} + \begin{bmatrix}\mathbf{X}^{\mathrm{T}} & \mathbf{x}\end{bmatrix}\begin{bmatrix}\mathbf{X} \\ \mathbf{x}^{\mathrm{T}}\end{bmatrix}\right)^{-1}}_{\mathbf{Q}} \begin{bmatrix}\mathbf{X}^{\mathrm{T}} & \mathbf{x}\end{bmatrix}\begin{bmatrix}\mathbf{y} \\ y\end{bmatrix} \tag{16}$$

with $\boldsymbol{\Sigma}_0 = \mathrm{diag}(\boldsymbol{\gamma}^{(P)}) \otimes \mathbf{B}^{(P)}$. Hence, the reward function for this estimator is given by (12) with $\mathbf{Q}$ from (16).

For best results, we need to initialize parameter $\boldsymbol{\Sigma}_0$ relative to signal power.

4 PROOF FOR THEOREM 1

Since this theorem focuses on single agent settings, both proposed TS algorithms in the theorem have the theoretical guarantees as well as the limits of TS bounds as summarized in [Russo et al., 2018, Ch. 8]. As discussed in this chapter, there are different theoretical approaches proposed in literature for regret bounds on TS. For this proof, we use regret bounds computed via information theory proposed in Russo and Van Roy [2016]. According to this paper, the resulting bounds better describe the benefits of prior distribution which helps point to failure modes of TS and potential solutions. Defining π_t as the distribution of actions for an online optimization policy at time t, Russo and Van Roy [2016] shows that for this policy, the expected regret defined in Theorem 1 is bounded by:

$$\mathbb{E}[\mathrm{Reg}(T)] \leq \sqrt{T\bar{\Psi}_t(\pi)\mathrm{H}(\mathbf{x}^\star)}, \tag{17}$$

where, $\mathrm{H}(\mathbf{x}^\star)$ is the entropy for optimal action $\mathbf{x}^\star$ and $\bar{\Psi}_t(\pi)$ is the average expected information ratio defined as follows.

$$\bar{\Psi}_t(\pi) = \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{\pi_t}[\Psi_t(\pi_t)] \quad \& \quad \Psi_t(\pi_t) \triangleq \frac{\Delta_t(\pi_t)^2}{\mathcal{I}_t(\pi_t)}, \tag{18}$$

where, $\Delta_t(\pi_t)$ and $\mathcal{I}_t(\pi_t)$ are the single-period expected regret and information gain over policy π_t at time step t defined below. For the single-period expected regret we have:

$$\Delta_t(\pi_t) = \sum_{\mathbf{x}\in\mathcal{X}} \pi(\mathbf{x})\,\Delta_t(\mathbf{x}) \quad \& \quad \Delta_t(\mathbf{x}) = \mathbb{E}[\mathcal{R}_t(\mathbf{x}^\star, \boldsymbol{\beta}) - \mathcal{R}_t(\mathbf{x}, \boldsymbol{\beta})]\,, \tag{19}$$

where, $\mathcal{R}_t(\mathbf{x}, \boldsymbol{\beta})$ is the reward function for action $\mathbf{x}$ at time t. And, for information gain, we have:

$$\mathcal{I}_t(\pi_t) = \sum_{\mathbf{x}\in\mathcal{X}} \pi(\mathbf{x})\,\mathcal{I}_t(\mathbf{x}^\star; y|\mathbf{x}), \tag{20}$$

where, $\mathcal{I}_t(\mathbf{x}^\star; y|\mathbf{x})$ is the mutual information between optimal action $\mathbf{x}^\star$ and observation $y = (\mathbf{x}^T\boldsymbol{\beta})^2$ at time t.

To compute the regret bound in (17), we need to first focus on computing single-period expected regret and information gain in (19) and (20). Computing these terms for a general problem formulation can be difficult. However, for an online optimization problem with a finite prior distribution and finite action set, [Russo and Van Roy, 2017, Algorithm 1] has provided the necessary formulation to compute these terms. Since our problem formulation falls under the category of this framework, we can take advantage of their formulations as follows. For single-period expected regret, we have:

$$\Delta_t(\mathbf{x}) = \mathcal{R}(\mathbf{x}^\star) - \sum_{\boldsymbol{\beta}} p(\boldsymbol{\beta})\mathcal{R}(\mathbf{x}, \boldsymbol{\beta}), \tag{21}$$

and, for the mutual information we have:

$$\mathcal{I}_t(\mathbf{x}^\star; y|\mathbf{x}) = \sum_{\mathbf{x}^\star, y} p(\mathbf{x}^\star, y|\mathbf{x}) \log\left(\frac{p(\mathbf{x}^\star, y|\mathbf{x})}{p(\mathbf{x}^\star)p(y|\mathbf{x})}\right), \tag{22}$$

where,

$$\begin{aligned} p(\mathbf{x}^\star) &= \sum_{\boldsymbol{\beta}\in\mathcal{B}(\mathbf{x}^\star)} p(\boldsymbol{\beta}|\mathbf{D}_t^1), \\ p(y|\mathbf{x}) &= \sum_{\boldsymbol{\beta}} p(\boldsymbol{\beta}|\mathbf{D}_t^1)p(y|\mathbf{x}, \boldsymbol{\beta}), \\ p(\mathbf{x}^\star, y|\mathbf{x}) &= p(\mathbf{x}^\star) \sum_{\boldsymbol{\beta}\in\mathcal{B}(\mathbf{x}^\star)} p(y|\mathbf{x}, \boldsymbol{\beta}), \end{aligned} \tag{23}$$

with $\mathcal{B}(\mathbf{x}) = \{\boldsymbol{\beta} : \mathbf{x} = \arg\max_{\tilde{\mathbf{x}}} \mathbb{E}_{y|\tilde{\mathbf{x}},\boldsymbol{\beta}}[\mathcal{R}(\tilde{\mathbf{x}}, \boldsymbol{\beta})]\}$ as the set containing all vectors $\boldsymbol{\beta}$ for which action $\mathbf{x}$ is optimal.

Next, we will compute this regret bound for our TS algorithms with 1-sparse and block sparse prior introduced in Theorem 1. Our emphasis here is on understanding the benefits of a block sparse prior distribution. Hence, to simplify the discussion we assume both algorithms are aware of the true 1-sparse prior for $\boldsymbol{\beta}$.

1-SPARSE PRIOR

Let us here establish two parameters $i^\star$ and $\tilde{i}$ that we will use throughout the proof. 1) Since the true parameter $\boldsymbol{\beta}$ is 1-sparse, we assume the location of the non-zero element of true $\boldsymbol{\beta}$ is $i^\star$. Given the uniform 1-sparse prior on true $\boldsymbol{\beta}$, $i^\star$ can be either of indices 1 through n with probability $1/n$. 2) Since our TS algorithm here assumes a 1-sparse prior distribution, the only feasible sensing actions in $\mathcal{X}$ are 1-sparse with one nonzero element at location $\tilde{i}$.

With two parameters $i^\star$ and $\tilde{i}$, we can now compute the single-period expected regret in (21) for $t = 1$ as:

$$\Delta_1(\mathbf{x}) = \Delta_1(\tilde{i}) = 1 - \sum_{i^\star=1}^{n} 1/n \times \mathbb{1}(\tilde{i} = i^\star),$$

Hence, the single-period expected regret $\Delta_1(\pi_1)$ over our policy becomes:

$$\Delta_1(\pi_1) = \sum_{\mathbf{x}\in\mathcal{X}} \pi(\mathbf{x})\,\Delta_1(\mathbf{x}) \overset{(a)}{=} \sum_{\tilde{i}=1}^{n} 1/n\,\Delta_1(\tilde{i}) = \frac{n-1}{n}$$

Here, (a) follows from the fact that our TS algorithm at $t = 1$ will pick all sensing actions with the nonzero element $j = 1$ through $j = n$ equally likely. Next, we need to compute single-period expected regret in (21) for $t = 2$. For $t = 2$, there are two policies for our TS algorithm depending on the action $\mathbf{x}_1$ chosen in previous time step. If $\mathbf{x}_1 = \mathbf{x}^\star$, then our TS algorithm will be exploiting the same optimal action at time $t = 2$ and set $\mathbf{x}_2 = \mathbf{x}^\star$ for which $\Delta_2(\mathbf{x}) = \Delta_2(\pi_2) = 0$. The probability of finding $\mathbf{x}^\star$ at $t = 1$ is $1/n$. However, if $\mathbf{x}_1 \neq \mathbf{x}^\star$, then our TS algorithm will be exploring the $n - 1$ feasible actions that are not observed. Let us assume $\tilde{i}_1$ shows the location of nonzero element in $\mathbf{x}_1$, then:

$$\Delta_2(\mathbf{x}) = \Delta_2(\tilde{i}) = 1 - \sum_{i^\star=1, i^\star\neq\tilde{i}_1}^{n} \frac{1}{n-1} \times \mathbb{1}(\tilde{i} = i^\star),$$

Hence, for both policies the single-period expected regret $\Delta_2(\pi_2)$ becomes:

$$\Delta_2(\pi_2) = \begin{cases} 0, & \text{prob} = \frac{1}{n} \\ \sum_{\mathbf{x}\in\mathcal{X}} \pi(\mathbf{x})\,\Delta_2(\mathbf{x}) = \sum_{\tilde{i}=1,\tilde{i}\neq\tilde{i}_1}^{n} \frac{1}{n-1}\,\Delta_2(\tilde{i}) = \frac{n-2}{n-1}, & \text{prob} = \frac{n-1}{n} \end{cases}$$

Similarly, we can conclude that for any $t > 2$, we will have:

$$\Delta_t(\pi_t) = \begin{cases} 0, & \text{prob} = \frac{t-1}{n} \\ \frac{n-t}{n-t+1}, & \text{prob} = \frac{n-t+1}{n} \end{cases} \tag{24}$$

We will next compute the information gain in (22) for time step t. Recall that our TS algorithm have two policies. One policy π_t is to keep exploiting the optimal action if it has been discovered in the previous $t-1$ measurements $\mathbf{D}_t^1$. Since $\Delta_t(\pi_t) = 0$ for this policy, we have $\Psi_t(\pi_t) = \frac{\Delta_t(\pi_t)^2}{\mathcal{I}_t(\pi_t)} = 0$ and hence there is no need to compute the information term $\mathcal{I}_t(\pi_t)$ for this scenario. The second policy of our TS method is used if optimal action $\mathbf{x}^\star$ has not been found in the previous $t-1$ measurements. For this scenario, at time t the TS algorithm picks the sensing action $\mathbf{x}_t$ out of $n-t+1$ measurements that have not been observed yet. For this policy, we can compute the probabilities in (23) as follows. First, for $p(\mathbf{x}^\star)$, we have:

$$p(\mathbf{x}^\star) = \sum_{\boldsymbol{\beta}\in\mathcal{B}(\mathbf{x}^\star)} p(\boldsymbol{\beta} = \mathbf{x}^\star|\mathbf{D}_t^1) = \begin{cases} \frac{1}{n-t+1}, & \text{if } \mathbf{x}^\star \notin \mathbf{D}_t^1 \ \wedge \ \mathbf{x}^\star \in \mathcal{X} \\ 0, & \text{otherwise} \end{cases}$$

Second, for $p(y|\mathbf{x})$ we have:

$$\begin{aligned} p(y|\mathbf{x}) = p(y|\tilde{i}) &= \begin{cases} \sum_{i^\star=1,i^\star\neq\{\tilde{i}_1,\ldots,\tilde{i}_{t-1}\}}^{n} \frac{1}{n-t+1}\ p(i^\star \neq \tilde{i}), & \text{if } y = 0, \\ \sum_{i^\star=1,i^\star\neq\{\tilde{i}_1,\ldots,\tilde{i}_{t-1}\}}^{n} \frac{1}{n-t+1}\ p(i^\star = \tilde{i}), & \text{if } y = 1, \end{cases} \\ &= \begin{cases} \frac{n-t}{n-t+1} & \text{if } y = 0, \\ \frac{1}{n-t+1} & \text{if } y = 1, \\ 0 & \text{otherwise}, \end{cases} \end{aligned}$$

where, $\{\tilde{i}_1, ..., \tilde{i}_{t-1}\}$ denotes the location of the nonzero elements in the previous $t-1$ measurements $\mathbf{D}_t^1$. Third, for $p(\mathbf{x}^\star, y|\mathbf{x})$ we get:

$$p(\mathbf{x}^\star, y|\mathbf{x}) = \frac{1}{n-t+1} \sum_{\boldsymbol{\beta}=\mathbf{x}^\star} p(y|\mathbf{x}, \boldsymbol{\beta}) = \begin{cases} \frac{1}{n-t+1} & \text{if } \ y = 0 \ \wedge \ \mathbf{x}^\star \neq \mathbf{x}, \\ \frac{1}{n-t+1} & \text{if } \ y = 1 \ \wedge \ \mathbf{x}^\star = \mathbf{x}, \\ 0 & \text{otherwise.} \end{cases}$$

Using these three probability densities, the information gain in (22) becomes:

$$\begin{aligned} \mathcal{I}_t(\mathbf{x}^\star; y|\mathbf{x}) = &\sum_{\substack{\mathbf{x}^\star\in\mathcal{X},\mathbf{x}^\star\notin\mathbf{D}_t^1 \\ \mathbf{x}^\star\neq\mathbf{x},y=0}} \frac{1}{n-t+1} \log\left(\frac{\frac{1}{n-t+1}}{\frac{1}{n-t+1}\times\frac{n-t}{n-t+1}}\right) \\ &+ \sum_{\substack{\mathbf{x}^\star\in\mathcal{X},\mathbf{x}^\star\notin\mathbf{D}_t^1 \\ \mathbf{x}^\star=\mathbf{x},y=1}} \frac{1}{n-t+1} \log\left(\frac{\frac{1}{n-t+1}}{\frac{1}{n-t+1}\times\frac{1}{n-t+1}}\right) \\ &= \frac{n-t}{n-t+1}\log\left(\frac{n-t+1}{n-t}\right) + \frac{1}{n-t+1}\log\left(n-t+1\right), \end{aligned}$$

resulting in:

$$\mathcal{I}_t(\pi_t) = \sum_{\mathbf{x}\in\mathcal{X},\mathbf{x}^\star\notin\mathbf{D}_t^1} \frac{1}{n-t+1}\,\mathcal{I}_t(\mathbf{x}^\star; y|\mathbf{x}) = \mathcal{I}_t(\mathbf{x}^\star; y|\mathbf{x}). \tag{25}$$

Now, using (24) and (25) , we can compute the average expected information ratio in (18) as:

$$\bar{\Psi}_t(\pi) = \frac{1}{T}\sum_{t=1}^{T}\left[\frac{t-1}{n}\times 0 + \frac{n-t+1}{n}\times\frac{\left(\frac{n-t}{n-t+1}\right)^2}{\frac{n-t}{n-t+1}\log\left(\frac{n-t+1}{n-t}\right) + \frac{1}{n-t+1}\log\left(n-t+1\right)}\right],$$

which together with an upperbound $\mathrm{H}(\mathbf{x}^\star) < \log(|\mathcal{X}|)$ and (17) gives the following regret bound:

$$\mathbb{E}[\mathrm{Reg}(T)] \leq \left(\log(|\mathcal{X}|) \sum_{t=1}^{T} \frac{\frac{(n-t)^2}{n(n-t+1)}}{\frac{n-t}{n-t+1}\log\left(\frac{n-t+1}{n-t}\right) + \frac{1}{n-t+1}\log(n-t+1)} \right)^{1/2}.$$

Lastly, the regret bound we computed above is only applicable when $T \leq n$. Since there are n 1-sparse actions available for a vector $\boldsymbol{\beta}$ with length n, after $T = n$ actions the algorithm has surely found the optimal action. As a result, $\Delta_t(\pi_t) = 0$ for $t \geq n$ which gives us the regret bound:

$$\mathbb{E}[\mathrm{Reg}(T)] \leq \left(\log(|\mathcal{X}|) \sum_{t=1}^{\min\{T,n-1\}} \frac{\left(1-\frac{t}{n}\right)\left(1-\frac{1}{n-t+1}\right)}{\left(\frac{n-t-1}{n-t}\log\left(\frac{n-t}{n-t-1}\right)+\frac{1}{n-t}\log(n-t)\right)} \right)^{1/2} \tag{26}$$

BLOCK SPARSE PRIOR

We will again start with establishing two parameters $i^\star$ and m_t that we will use throughout the proof. 1) We assume the location of the non-zero element of true $\boldsymbol{\beta}$ is $i^\star$. Given the uniform 1-sparse prior on true $\boldsymbol{\beta}$, $i^\star$ can be either of indices 1 through n with probability $1/n$. 2) Since our TS algorithm here assumes a 1-block sparse prior distribution, the feasible sensing actions in $\mathcal{X}$ at time step t are $\mathbf{x} = [\gamma_1 \mathbf{b}, ..., \gamma_{M_t}\mathbf{b}]^{\mathrm{T}}$ where only one of paramaters γ_1 through γ_{M_t} is nonzero and $\mathbf{b} = [\frac{1}{\sqrt{L_t}}, ..., \frac{1}{\sqrt{L_t}}]$ are the blocks with length L_t. We call the location of the nonzero block m_t which can be either of indices 1 through $M_t = n/L_t$ with probability $1/M_t$ at time step t.

With two parameters $i^\star$ and m_t, we can now compute the single-period expected regret $\Delta_1(\pi_1)$ in (19) for $t = 1$ as:

$$\begin{aligned}
\Delta_1(\pi_1) &= \sum_{\mathbf{x}\in\mathcal{X}} \pi(\mathbf{x})\,\Delta_1(\mathbf{x}) \overset{(b)}{=} \sum_{m_1=1}^{M_1} \frac{1}{M_1}\,\Delta_1(m_1) \\
&\overset{(c)}{=} \sum_{m_1=1}^{M_1} \frac{1}{M_1}\left(1 - \sum_{i^\star=1}^{n} 1/n \times \mathbb{1}(i^\star \in \{m_1L_1+1, ..., m_1L_1+L_1\}) \times \frac{1}{L_1}\right) = \frac{n-1}{n},
\end{aligned}$$

here, (b) follows from the fact that our TS algorithm at $t = 1$ will pick all sensing actions with the nonzero blocks $m_1 = 1$ through $m_1 = M_1$ equally likely, and (c) follows from (21). For $t > 2$, there are two policies depending on whether $i^\star$ has been discovered to belong to any of the elements of block m_1 or not. If $i^\star$ does not belong to block m_1, then:

$$\begin{aligned}
\Delta_2(\pi_2) &= \sum_{\substack{\mathbf{x}\in\mathcal{X} \\ \mathbf{x}\notin\mathbf{D}_t^1}} \pi(\mathbf{x}) \left(1 - \sum_{\substack{i^\star=1 \\ i^\star\notin\{m_1L_1+1,...,m_1L_1+L_1\}}}^{n} \frac{1}{n-L_1} \times \mathbb{1}(i^\star \in \{m_2L_2+1, ..., m_2L_2+L_2\}) \times \frac{1}{L_2}\right) \\
&= 1 - \frac{1}{n-L_1}.
\end{aligned}$$

Or if $i^\star$ does belong to block m_1, similarly we $\Delta_2(\pi_2) = 1 - \frac{1}{L_1}$. For block length $L_1 = n/2$, we will have the regret $\Delta_2(\pi_2) = 1 - \frac{1}{n/2}$ for both policies which is the smallest regret value. Similarly, for any $t > 2$, we have:

$$\Delta_t(\pi_t) = 1 - \frac{1}{n - \sum_{t'=1}^{t-1} \frac{n}{2^{t'}}}, \tag{27}$$

as long as $L_t = \frac{n}{2^t}$.

We will next compute the information gain in (22) for time step $t = 1$. For that, we will first compute the three probability densities in (23). For $p(\mathbf{x}^\star)$, we have:

$$p(\mathbf{x}^\star) = \sum_{\boldsymbol{\beta}\in\mathcal{B}(\mathbf{x}^\star)} p(\boldsymbol{\beta} = \mathbf{x}^\star) = \begin{cases} \frac{1}{n}, & \text{if } \mathbf{x}^\star \text{ is 1-sparse} \\ 0, & \text{otherwise} \end{cases}$$

which follows the fact that the optimal action is 1-sparse. Second, for $p(y|\mathbf{x})$ we have:

$$p(y|\mathbf{x}) = p(y|m_1) = \begin{cases} \sum_{i^\star=1}^{n} \frac{1}{n}\ p(i^\star \notin \{m_1L_1+1,...,m_1L_1+L_1\}), & \text{if } y=0, \\ \sum_{i^\star=1}^{n} \frac{1}{n}\ p(i^\star \in \{m_1L_1+1,...,m_1L_1+L_1\}), & \text{if } y=1/L_1, \end{cases}$$
$$= \begin{cases} \frac{n-L_1}{n} & \text{if } y=0, \\ \frac{L_1}{n} & \text{if } y=1/L_1, \\ 0 & \text{otherwise,} \end{cases}$$

Third, for $p(\mathbf{x}^\star, y|\mathbf{x})$ we get:

$$p(\mathbf{x}^\star, y|\mathbf{x}) = \frac{1}{n}\sum_{\boldsymbol{\beta}=\mathbf{x}^\star} p(y|\mathbf{x},\boldsymbol{\beta}) = \begin{cases} \frac{1}{n} & \text{if } y=0 \ \wedge\ \mathbf{x}^{\mathrm{T}}\mathbf{x}^\star = 0, \\ \frac{1}{n} & \text{if } y=1/L_1 \ \wedge\ \mathbf{x}^{\mathrm{T}}\mathbf{x}^\star \neq 0, \\ 0 & \text{otherwise.} \end{cases}$$

Using these three probability densities, the information gain in (22) becomes:

$$\mathcal{I}_1(\mathbf{x}^\star; y|\mathbf{x}) = \sum_{\substack{\mathbf{x}^\star\in\mathcal{X} \\ \mathbf{x}^{\mathrm{T}}\mathbf{x}^\star\neq 0, y=1/L_1}} \frac{1}{n}\log\left(\frac{\frac{1}{n}}{\frac{1}{n}\times\frac{L_1}{n}}\right) + \sum_{\substack{\mathbf{x}^\star\in\mathcal{X} \\ \mathbf{x}^{\mathrm{T}}\mathbf{x}^\star= 0, y=0}} \frac{1}{n}\log\left(\frac{\frac{1}{n}}{\frac{1}{n}\times\frac{n-L_1}{n}}\right)$$
$$= \frac{L_1}{n}\log\left(\frac{n}{L_1}\right) + \frac{n-L_1}{n}\log\left(\frac{n}{n-L_1}\right).$$

The information term above is maximized when $L_1 = n/2$ which gives $\mathcal{I}_1(\mathbf{x}^\star; y|\mathbf{x}) = \log(2)$.

Next, let us discuss information gain for $t=2$. Recall that for $t=2$, there are two policies depending on whether $i^\star$ has been discovered to belong to elements of block m_1 or not. Similar computations to that of $t=1$ shows that for the two policies we have:

$$\mathcal{I}_2(\mathbf{x}^\star; y|\mathbf{x}) = \begin{cases} \frac{L_2}{L_1}\log\left(\frac{L_1}{L_2}\right) + \frac{L_1-L_2}{L_1}\log\left(\frac{L_1}{L_1-L_2}\right), & \text{if } i^\star \in \{m_1L_1+1,...,m_1L_1+L_1\} \\ \frac{L_2}{n-L_1}\log\left(\frac{n-L_1}{L_2}\right) + \frac{n-L_1-L_2}{n-L_1}\log\left(\frac{n-L_1}{n-L_1-L_2}\right), & \text{if } i^\star \notin \{m_1L_1+1,...,m_1L_1+L_1\} \end{cases}$$

By setting $L_1 = n/2$ and $L_2 = n/4$, both policies will be maximized to give $\mathcal{I}_2(\mathbf{x}^\star; y|\mathbf{x}) = \log(2)$. Similarly, for any $t>2$, the information gain $\mathcal{I}_t(\mathbf{x}^\star; y|\mathbf{x})$ is maximized with:

$$\mathcal{I}_t(\mathbf{x}^\star; y|\mathbf{x}) = \log(2), \tag{28}$$

as long as $L_t = \frac{n}{2^t}$. The fact that the varying block length $L_t = \frac{n}{2^t}$ maximizes information gain $\mathcal{I}_t(\mathbf{x}^\star; y|\mathbf{x})$ and minimizes regret term $\Delta_t(\pi_t)$ shows that this varying block length is the optimal approach for our TS algorithm finding a 1-sparse signal of interest. This result in essence describes our strategy for varying block length $L_t = \frac{L_{t-1}}{2}$in SPATS algorithm.

Finally, using (27) and (28) with the upperbound $\mathrm{H}(\mathbf{x}^\star) < \log(|\mathcal{X}|)$, we can compute the expected regret in (17) as:

$$\mathbb{E}[\mathrm{Reg}(T)] \leq \left(\log(|\mathcal{X}|)\sum_{t=1}^{T}\left[\frac{\left(1-\frac{1}{n-\sum_{t'=1}^{t-1}\frac{n}{2^{t'}}}\right)^2}{\log(2)}\right]\right)^{1/2},$$

Lastly, the regret bound we computed above is only applicable when $T \leq \log_2(n)$. Since the algorithm has a varying block length of $L_t = \frac{n}{2^t}$, after $T = \log_2(n)$ the block length is reduced to 1 which certainly includes nonzero element of $\boldsymbol{\beta}$. As a result, after $T = \log_2(n)$ actions the algorithm has surely found the optimal action. This result means that $\Delta_t(\pi_t) = 0$ for $t > \log_2(n)$ which gives us the regret bound:

$$\mathbb{E}[\mathrm{Reg}(T)] \leq \left(\log(|\mathcal{X}|)\sum_{t=1}^{\min\{T,\log_2(n)\}}\left(1-\frac{1}{n-\left(\sum_{t'=1}^{t-1}\frac{n}{2^{t'}}\right)}\right)^2/\log(2)\right)^{1/2}. \tag{29}$$

5 PROOF FOR THEOREM 2

Unlike Theorem 1, we use a direct approach to compute expected regret in this proof. Similar to Appendix 4, to simplify the discussion, we assume both algorithms are aware of the prior distribution on true parameter $\boldsymbol{\beta}$. We will first compute the expected regret for the single agent algorithm.

SINGLE AGENT

Since the true parameter $\boldsymbol{\beta}$ is 1-sparse, we assume the location of the non-zero element is $i^\star$. Recall the equation for expected regret:

$$\mathbb{E}[\text{Reg(T)}] = \mathbb{E}_{i^\star, \mathbf{x}_1, \ldots, \mathbf{x}_T}\left[\sum_{t=1}^{T}\left[\mathcal{R}(\mathbf{x}^\star, \boldsymbol{\beta}) - \mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta})\right]\right] \overset{(a)}{=} \sum_{t=1}^{T}\mathcal{R}(\mathbf{x}^\star, \boldsymbol{\beta}) - \sum_{t=1}^{T}\mathbb{E}_{\mathbf{x}_t}\left[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta})\right], \tag{30}$$

where, (a) follows from assuming $i^\star$ is a fixed unknown location without loss of generality. To compute the first term $\mathcal{R}(\mathbf{x}^\star, \boldsymbol{\beta})$, note that the optimal action $\mathbf{x}^\star$ is the 1-sparse vector with $i^\star$ as the nonzero element, i.e. $\mathcal{R}(\mathbf{x}^\star, \boldsymbol{\beta}) = 1$.

To compute the second term $\mathbb{E}_{\mathbf{x}_t}[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta})]$, remember that $\mathbf{x}_t$ for TS is selected by maximizing the reward function given the available measurements $\mathbf{D}_t^1$. For a single agent setting, as depicted in Figure 2a, we have $\mathbf{D}_t^1 = \{(\mathbf{x}_{t'}, y_{t'}) | t' = 1, .., t-1\}$. Since our algorithm has a 1-sparse prior, only feasible sensing actions are 1-sparse. If $\mathbf{x}^\star \in \mathbf{D}_t^1$, then TS will be exploiting the same optimal action at time t and set $\mathbf{x}_t = \mathbf{x}^\star$ for which $\mathbb{E}_{\mathbf{x}_t}\left[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}^\star \in \mathbf{D}_t^1\right] = \mathbb{E}_{\mathbf{x}_t}[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}_t = \mathbf{x}^\star] = 1$. On the other hand, if $\mathbf{x}^\star \notin \mathbf{D}_t^1$, then TS will select action $\mathbf{x}_t$ out of $n - t + 1$ actions that are not observed as part of $\mathbf{D}_t^1$. Hence, the reward $\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta})$ is 1 with probability $\frac{1}{n-t+1}$ and 0 otherwise, i.e $\mathbb{E}_{\mathbf{x}_t}\left[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}^\star \notin \mathbf{D}_t^1\right] = \frac{1}{n-t+1}$.

Putting the information for the first and second term in (30), we have:

$$\begin{aligned}
\mathbb{E}[\text{Reg(T)}] &= \sum_{t=1}^{T}\mathcal{R}(\mathbf{x}^\star, \boldsymbol{\beta}) - \sum_{t=1}^{T}\mathbb{E}_{\mathbf{x}_t}[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta})] \\
&= T - \sum_{t=1}^{T}\Big[p(\mathbf{x}^\star \in \mathbf{D}_t^1) \times \mathbb{E}_{\mathbf{x}_t}\left[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}^\star \in \mathbf{D}_t^1\right] \\
&\qquad + p(\mathbf{x}^\star \notin \mathbf{D}_t^1) \times \mathbb{E}_{\mathbf{x}_t}\left[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}^\star \notin \mathbf{D}_t^1\right]\Big] \\
&\overset{(b)}{=} T - \sum_{t=1}^{T}\Big[\frac{t-1}{n} \times 1 + \frac{n-t+1}{n} \times \frac{1}{n-t+1}\Big] \\
&= T - \frac{T(T+1)}{2n},
\end{aligned}$$

where, (b) follows from $p(\mathbf{x}^\star \in \mathbf{D}_t^1) = 1 - p(\mathbf{x}^\star \notin \mathbf{D}_t^1) = \frac{t-1}{n}$ given that there are $t-1$ distinct measurements out of n distinct possible actions until $\mathbf{x}^\star$ is found.

Lastly, the regret term we computed above is only applicable when $T \leq n$. Since there are n 1-sparse actions available for a vector $\boldsymbol{\beta}$ with length n, after $T = n$ actions the algorithm has surely found the optimal action. As a result, $\mathcal{R}(\mathbf{x}^\star, \boldsymbol{\beta}) - \mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) = 1 - 1 = 0$ for $t > n$ which gives us:

$$\mathbb{E}[\text{Reg}(T)] = T_n - \frac{T_n(T_n+1)}{2n}, \quad T_n = \min\{T, n\}.$$

MULTI-AGENT

Now, we will compute the expected regret for TS with asynchronous multi-agent setting. Similar to single agent, the equation for expected regret follows from (30) with $i^\star$ as the location of the nonzero element of $\boldsymbol{\beta}$. Again, for the first term we have $\mathcal{R}(\mathbf{x}^\star, \boldsymbol{\beta}) = 1$ where the optimal action $\mathbf{x}^\star$ is the 1-sparse vector with $i^\star$ as the nonzero element. For the second term $\mathbb{E}_{\mathbf{x}_t}[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta})]$, we know that if $\mathbf{x}^\star \in \mathbf{D}_t^j$, then TS will be exploiting the same optimal action at time t and set $\mathbf{x}_t = \mathbf{x}^\star$ for which $\mathbb{E}_{\mathbf{x}_t}\left[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}^\star \in \mathbf{D}_t^j\right] = 1$. On the other hand, if $\mathbf{x}^\star \notin \mathbf{D}_t^j$, then TS will select action $\mathbf{x}_t$ out of actions that are not

observed as part of $\mathbf{D}_t^j$. Precisely:

$$\mathbb{E}[\text{Reg(T)}] = T - \sum_{t=1}^{T} \Big[p(\mathbf{x}^\star \in \mathbf{D}_t^j) \times \mathbb{E}_{\mathbf{x}_t}\Big[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}^\star \in \mathbf{D}_t^j\Big] + p(\mathbf{x}^\star \notin \mathbf{D}_t^j) \times \mathbb{E}_{\mathbf{x}_t}\Big[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}^\star \notin \mathbf{D}_t^j\Big] \Big]. \tag{31}$$

However, computing $p(\mathbf{x}^\star \in \mathbf{D}_t^j) = 1 - p(\mathbf{x}^\star \notin \mathbf{D}_t^j)$ and $\mathbb{E}_{\mathbf{x}_t}\Big[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}^\star \notin \mathbf{D}_t^j\Big]$ is not as straight forward as the single agent setting. In a single agent setting, $|\mathbf{D}_t^j| = t-1$ and all $t-1$ actions are distinct until $\mathbf{x}^\star$ is found. In an asynchronous multi-agent setting, $|\mathbf{D}_t^j| < t-1$ and some of the actions in $\mathbf{D}_t^j$ might be the same. Specifically, as illustrated in Figure 2b an action t starts before all previous $t-1$ actions are completed and as a result there is a chance that action t will equal the previous actions that were not included in $\mathbf{D}_t^j$. While we cannot precisely compute $p(\mathbf{x}^\star \in \mathbf{D}_t^j)$ in this asynchronous multi-agent setting, in the following lemma we compute a lower bound for it with the proof provided in Appendix 5.1.

Lemma 1. *Consider the active search problem in Theorem 2. For an asynchronous multi-agent TS algorithm,* $p(\mathbf{x}^\star \in \mathbf{D}_t^j)$ *is bounded by:*

$$p(\mathbf{x}^\star \in \mathbf{D}_t^j) \geq 1 - \frac{(n-1)^t}{n^t}, \quad \textit{if } \ t < J$$
$$p(\mathbf{x}^\star \in \mathbf{D}_t^j) \geq 1 - \frac{(n-1)^{J-1}(n-t+2J-1)}{n^J}, \quad \textit{if } \ t \geq J$$

Using the bound in Lemma 1 along with a naïve bound $\mathbb{E}_{\mathbf{x}_t}\Big[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}^\star \notin \mathbf{D}_t^j\Big] \geq 0$, we can upper bound the expected regret in (31) as follows:

$$\begin{aligned}
\mathbb{E}[\text{Reg(T)}] &\leq T - \sum_{t=1}^{T} \Big[p(\mathbf{x}^\star \in \mathbf{D}_t^j) \times \mathbb{E}_{\mathbf{x}_t}\Big[\mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) | \mathbf{x}^\star \in \mathbf{D}_t^j\Big] \Big] \\
&= T - \sum_{t=1}^{J} \Big[1 - \frac{(n-1)^t}{n^t} \times 1\Big] - \sum_{t=J+1}^{T} \Big[1 - \frac{(n-1)^{J-1}(n-t+2J-1)}{n^J} \times 1\Big] \\
&\leq T - J + \sum_{t=1}^{J} \Big[\frac{(n)^t}{n^t}\Big] - (T-J) + \sum_{t=J+1}^{T} \Big[\frac{(n)^{J-1}(n-t+2J-1)}{n^J}\Big] \\
&= T - \frac{T(T+1)}{2n} + \frac{T(2J-1)}{n} - \frac{3J^2-3J}{2n} \\
&\leq T - \frac{T(T+1)}{2n} + \frac{T(2J-1)}{n}.
\end{aligned}$$

Lastly, the regret term we computed above is only applicable when $T \leq n+J$. If $T > n+J$, then $|\mathbf{D}_t^j| = n$ which includes all n 1-sparse actions available for a vector $\boldsymbol{\beta}$ with length n. Hence, after $T = n+J$ actions, the algorithm has surely found the optimal action which results in $\mathcal{R}(\mathbf{x}^\star, \boldsymbol{\beta}) - \mathcal{R}(\mathbf{x}_t, \boldsymbol{\beta}) = 1 - 1 = 0$, and consequently:

$$\mathbb{E}[\text{Reg}(T)] \leq T_n - \frac{T_n(T_n+1)}{2n} + \frac{T_n(2J-1)}{n}, \quad T_n = \min\{T, n+J\}$$

5.1 PROOF FOR LEMMA 1

Proof. Since we need to upper bound the expected regret in (30), we will lower bound $p(\mathbf{x}^\star \in \mathbf{D}_t^j)$. To compute this bound, we need to better understand $\mathbf{D}_t^j$. For that, we will look at the example illustrated in Figure 2b. For this example, $\mathbf{D}_1^1$ through $\mathbf{D}_{12}^1$ have the following measurement sets:

$$\mathbf{D}_1^1 = \emptyset \qquad \mathbf{D}_7^3 = \{(\mathbf{x}_{t'}, y_{t'}) | t' = \{1,3,2,5\}\}$$
$$\mathbf{D}_2^2 = \emptyset \qquad \mathbf{D}_8^2 = \{(\mathbf{x}_{t'}, y_{t'}) | t' = \{1,3,2,5,6\}\}$$
$$\mathbf{D}_3^3 = \emptyset \qquad \mathbf{D}_9^1 = \{(\mathbf{x}_{t'}, y_{t'}) | t' = \{1,3,2,5,6,4\}\}$$

$$\mathbf{D}_4^1 = \{(\mathbf{x}_{t'}, y_{t'})|t' = \{1\}\} \qquad \mathbf{D}_{10}^2 = \{(\mathbf{x}_{t'}, y_{t'})|t' = \{1,3,2,5,6,4,8\}\}$$
$$\mathbf{D}_5^3 = \{(\mathbf{x}_{t'}, y_{t'})|t' = \{1,3\}\} \qquad \mathbf{D}_{11}^3 = \{(\mathbf{x}_{t'}, y_{t'})|t' = \{1,3,2,5,6,4,8,7\}\}$$
$$\mathbf{D}_6^2 = \{(\mathbf{x}_{t'}, y_{t'})|t' = \{1,3,2\}\} \qquad \mathbf{D}_{12}^1 = \{(\mathbf{x}_{t'}, y_{t'})|t' = \{1,3,2,5,6,4,8,7,9\}\}$$

Our first observation is that $|\mathbf{D}_t^j| = t - J$ which applies to any asynchronous example [Kandasamy et al., 2018]. Next, let us compute $p(\mathbf{x}^\star \in \mathbf{D}_t^j)$ as an example for $\mathbf{D}_7^3$:

$$\begin{aligned}
p(\mathbf{x}^\star \in \mathbf{D}_7^3) &= 1 - p(\mathbf{x}^\star \notin \mathbf{D}_7^3) = 1 - p\Big((\mathbf{x}^\star \neq \mathbf{x}_1) \wedge (\mathbf{x}^\star \neq \mathbf{x}_3) \wedge (\mathbf{x}^\star \neq \mathbf{x}_2) \wedge (\mathbf{x}^\star \neq \mathbf{x}_5)\Big) \\
&= 1 - \Big(p(\mathbf{x}^\star \neq \mathbf{x}_1) \times p(\mathbf{x}^\star \neq \mathbf{x}_3|\mathbf{x}^\star \neq \mathbf{x}_1) \times p(\mathbf{x}^\star \neq \mathbf{x}_2|\mathbf{x}^\star \neq \mathbf{x}_1, \mathbf{x}_3) \\
&\qquad \times p(\mathbf{x}^\star \neq \mathbf{x}_5|\mathbf{x}^\star \neq \mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_2)\Big) \\
&\overset{(c)}{=} 1 - \Big(p(\mathbf{x}^\star \neq \mathbf{x}_1) \times p(\mathbf{x}^\star \neq \mathbf{x}_3|\emptyset) \times p(\mathbf{x}^\star \neq \mathbf{x}_2|\emptyset) \times p(\mathbf{x}^\star \neq \mathbf{x}_5|\mathbf{x}^\star \neq \mathbf{x}_1, \mathbf{x}_3)\Big) \\
&\overset{(d)}{=} 1 - \frac{n-1}{n} \times \frac{n-1}{n} \times \frac{n-1}{n} \times \frac{n-3}{n-2}.
\end{aligned} \tag{32}$$

Here, (c) follows from the fact that on picking $\mathbf{x}_3$ and $\mathbf{x}_2$, they depend on empty sets of measurements $\mathbf{D}_3^3$ and $\mathbf{D}_2^2$. And, on choosing sensing action $\mathbf{x}_5$, the algorithm has access to measurements $\mathbf{D}_5^3 = \{(\mathbf{x}_{t'}, y_{t'})|t' = \{1,3\}\}$. Consequently, in (d) we see that $\mathbf{x}_1$,$\mathbf{x}_3$ and $\mathbf{x}_2$ have n possible actions to pick, while $\mathbf{x}_5$ will pick out of $n-2$ actions that had excluded the ones in $\mathbf{D}_5^3$.

As evident in (32), computing $p(\mathbf{x}^\star \in \mathbf{D}_t^j)$ depends on the specifics of every example and is different for all $\mathbf{D}_t^j$. However, this probability can be lower-bounded for any asynchronous setting as follows. We know that $\mathbf{D}_t^j$ has a cardinality of $t - J$. As a result, each asynchronous example translates to a set $\mathbf{D}_t^j$ with permutation of $t - J$ measurements picked out of all $t-1$ possible measurements $\{(\mathbf{x}_{t'}, y_{t'})|\{t' = 1, ..., t-1\}\}$. Out of all possible asynchronous examples, $\mathbf{D}_t^W = \{(\mathbf{x}_{t'}, y_{t'})|\{t' = 1, ..., t-J\}\}$ is the worst case scenario in terms of finding the optimal action. In other words, for this measurement set, the algorithm has used the least amount of information to decide on each of the actions $\mathbf{x}_1$ through $\mathbf{x}_t$. Hence, the asynchronous example with $\mathbf{D}_t^W$ has the lowest probability to pick the optimal sensing action $\mathbf{x}^\star$ and can be used to lower-bound $p(\mathbf{x}^\star \in \mathbf{D}_t^j)$:

$$p(\mathbf{x}^\star \in \mathbf{D}_t^j) \geq p(\mathbf{x}^\star \in \mathbf{D}_t^W)$$

Now, all we have to do is compute $p(\mathbf{x}^\star \in \mathbf{D}_t^W)$. For $t < J$, we have:

$$\begin{aligned}
p(\mathbf{x}^\star \in \mathbf{D}_t^W) &= 1 - p(\mathbf{x}^\star \notin \mathbf{D}_t^W) = 1 - \Big(p(\mathbf{x}^\star \neq \mathbf{x}_1) \times p(\mathbf{x}^\star \neq \mathbf{x}_2|\emptyset) \times ... \times p(\mathbf{x}^\star \neq \mathbf{x}_t|\emptyset)\Big) \\
&= 1 - \underbrace{\frac{n-1}{n} \times \frac{n-1}{n} \times ... \times \frac{n-1}{n}}_{t \text{ terms}} = 1 - \frac{(n-1)^t}{n^t}
\end{aligned}$$

For $t \geq J$, we have:

$$\begin{aligned}
p(\mathbf{x}^\star \in \mathbf{D}_t^W) &= 1 - p(\mathbf{x}^\star \notin \mathbf{D}_t^W) \\
&= 1 - \Big(p(\mathbf{x}^\star \neq \mathbf{x}_1) \times p(\mathbf{x}^\star \neq \mathbf{x}_2|\emptyset) \times ... \times p(\mathbf{x}^\star \neq \mathbf{x}_J|\emptyset) \\
&\qquad \times p(\mathbf{x}^\star \neq \mathbf{x}_{J+1}|\mathbf{x}^\star \neq \mathbf{x}_1) p(\mathbf{x}^\star \neq \mathbf{x}_{J+2}|\mathbf{x}^\star \neq \mathbf{x}_1, \mathbf{x}_2) \\
&\qquad \times ... \times p(\mathbf{x}^\star \neq \mathbf{x}_{t-J}|\mathbf{x}^\star \neq \mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{t-2J})\Big) \\
&= 1 - \underbrace{\frac{n-1}{n} \times \frac{n-1}{n} \times ... \times \frac{n-1}{n}}_{J \text{ terms}} \times \frac{n-2}{n-1} \times \frac{n-3}{n-2} \times ... \times \frac{n-(t-2J)-1}{n-(t-2J)} \\
&= 1 - \frac{(n-1)^{J-1}(n-t+2J-1)}{n^J}
\end{aligned}$$

■

6 ADDITIONAL DETAILS ON LATSI

In this section, we provide additional details on deriving LATSI algorithm we proposed in Section 4.2. First, let us do a short review on IDS and RSI algorithms:

REVIEW OF IDS (INFORMATION DIRECTED SAMPLING)

There are certain online optimization problems such as our active search problem where traditional multi-armed bandit algorithms such as TS and Upper Confidence Bound (UCB) fail due to a careless assessment of the information gain (see Appendix 2). IDS is an online optimization algorithm proposed by Russo and Van Roy [2017] that addresses this problem by introducing a novel reward function that balances between expected single-period regret and a measure of information gain. Specifically, defining π_t^{IDS} as the action sampling distribution of IDS at time t, at each time step t, IDS is computed by:

$$\pi_t^{\text{IDS}} = \underset{\pi \in \mathcal{D}(\mathcal{X})}{\arg\min} \left\{ \Psi_t(\pi) \triangleq \frac{\Delta_t(\pi)^2}{\mathcal{I}_t(\pi)} \right\},$$

where they call $\Psi_t(\pi)$ the information ratio of the action sampling distribution π, and $\Delta_t(\pi)$ and $\mathcal{I}_t(\pi)$ are the single-period expected regret and information gain at time t defined below. For the single-period expected regret, we have:

$$\Delta_t(\pi) = \sum_{\mathbf{x} \in \mathcal{X}} \pi(\mathbf{x})\, \mathbb{E}[\mathcal{R}_t(\mathbf{x}^\star) - \mathcal{R}_t(\mathbf{x})]\,,$$

where, $\mathcal{R}_t(\mathbf{x})$ is the reward function for action $\mathbf{x}$ at time t. And, for information gain, we have:

$$\mathcal{I}_t(\pi) = \sum_{\mathbf{x} \in \mathcal{X}} \pi(\mathbf{x}) \mathbf{I}_t(\mathbf{x}^\star; y|\mathbf{x}),$$

where, $\mathcal{I}_t(\mathbf{x}^\star; y|\mathbf{x})$ is the mutual information between optimal action $\mathbf{x}^\star$ and observations y at time t if sensing action $\mathbf{x}$ is chosen.

Once IDS computes the action sampling distribution at time t, it will choose an action by randomly sampling this distribution.

REVIEW OF RSI (REGION SENSING INDEX)

RSI is a single agent active search algorithm designed to locate sparse signals by actively making data-collection decisions. Similar to our problem formulation in Section 2, RSI makes a practical assumption that at each time step the agent senses a contiguous region of the space. To decide on their next action, RSI at each time step chooses the sensing action $\mathbf{x}_t$ that maximizes the mutual information between the next observation y_t and the signal of interest $\boldsymbol{\beta}$, i.e.

$$\mathbf{x}_t = \underset{\mathbf{x}}{\arg\max}\, I(\boldsymbol{\beta}; y|\mathbf{x}, \mathbf{D}_t^1),$$

where, the mutual information is computed using posterior distribution $p(\boldsymbol{\beta}|\mathbf{D}_t^1) = p_0(\boldsymbol{\beta}) \prod_{\tau=1}^{t-1} p(y_\tau|\mathbf{x}, \boldsymbol{\beta})$ with a k-sparse uniform prior $p_0(\boldsymbol{\beta})$ and same likelihood distribution as in (1).

Unfortunately, computing the mutual information $I(\boldsymbol{\beta}; y|\mathbf{x}, \mathbf{D}_t^1)$ has high complexity for sparsity rates of $k > 1$. In order to reduce this complexity for $k > 1$ RSI recovers the support of $\boldsymbol{\beta}$ by repeatedly applying RSI assuming $k = 1$.

DERIVING LATSI

Recall how IDS algorithm solves for the failure mode of TS by using a novel reward function that is the ratio of the expected single-period regret and information gain. Inspired by this algorithm, we propose revising the expected reward $\lambda^+(\boldsymbol{\beta}^\star, \mathbf{D}_t^j, \mathbf{x})$ for agent j in (12) for Laplace-TS by adding a measure of information gain to it. For the information gain, we use the mutual information $I(\boldsymbol{\beta}; y|\mathbf{x}, \mathbf{D}_t^j)$ computed in RSI algorithm. Thus, we can say that LATSI is the Laplace-TS algorithm where the design stage has been replaced by:

$$\mathbf{x}_t = \underset{\mathbf{x}}{\arg\max}\, \boldsymbol{R}^+(\boldsymbol{\beta}^\star, \mathbf{D}_t^j, \{\mathbf{x}, y\}),$$

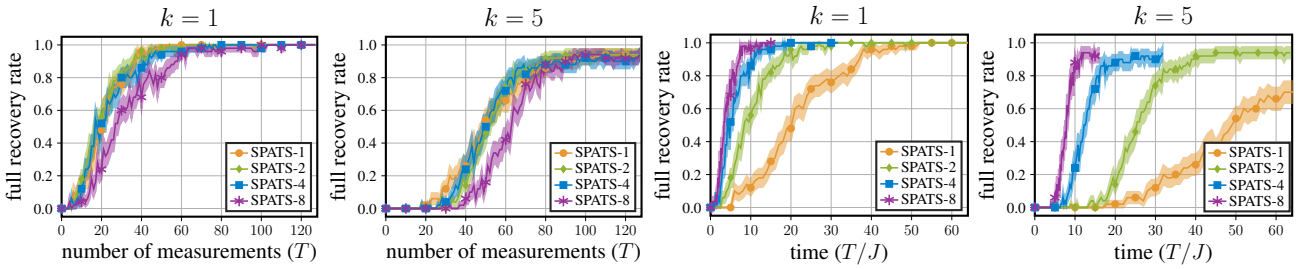


Figure 8: Full recovery rate of SPATS with 1, 2, 4 and 8 agents for $n = 128$, $k = 1, 5$

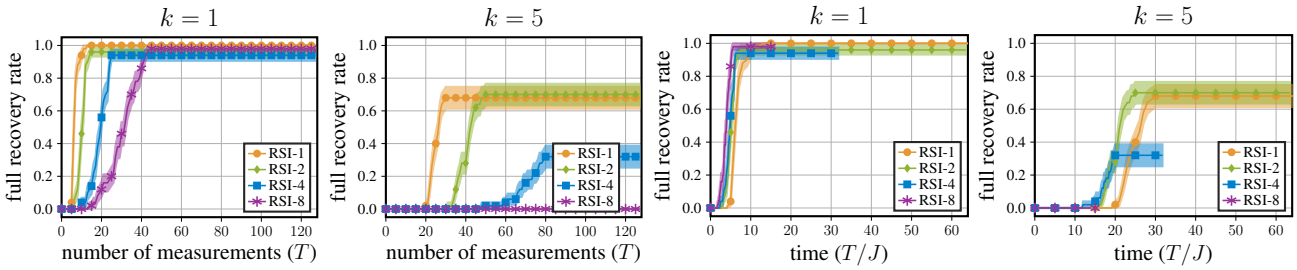


Figure 9: Full recovery rate of RSI with 1, 2, 4 and 8 agents for $n = 128$, $k = 1, 5$

where,

$$\boldsymbol{R}^{+} = \frac{I(\boldsymbol{\beta}^{\star}; y|\mathbf{x}, \mathbf{D}_t^j)}{\text{average } I \text{ over all } \mathbf{x}} + \alpha \times \frac{\lambda^{+}(\boldsymbol{\beta}^{\star}, \mathbf{D}_t^j, \mathbf{x})}{\text{average } \lambda^{+} \text{ over all } \mathbf{x}}, \tag{33}$$

using I from Ma et al. [2017] and λ^{+} in (12). Here, we normalize and add a tuning parameter α to best control the importance of each term on the overall reward $\boldsymbol{R}^{+}$.

7 ADDITIONAL NUMERICAL RESULTS

We now provide additional simulation results to support our analysis in Section 5.

1-DIMENSIONAL SEARCH SPACE

In this section we provide additional results for a 1-dimensional search space ($d = 1$), where we estimate a k-sparse signal β of length $n = 128$ with two sparsity rates of $k = 1, 5$. Here, we use the same set of parameters as those outlined in Section 5. Figure 8, 9 and 10 show the performance of SPATS, RSI and LATSI respectively. Our observations overall are similar to those in Section 5. Concretely, we observe that both SPATS and LATSI show an improvement in performance with time as more agents become available. The efficiency of this asynchronous search by SPATS and LATSI becomes more pronounced with higher sparsity rate k where there are more targets in the search space. On the other hand, multi-agent RSI's performance worsens as we increase the number of agents (especially with larger sparsity rate k). This observation is similar to that in Section 5 and is explained by the lack of randomness in the information theoretic reward function of RSI.

2-DIMENSIONAL SEARCH SPACE

In this section we provide additional results for a 2-dimensional search space ($d = 2$), where we estimate a k-sparse signal β of length $n = 16 \times 16$ and two sparsity rates of $k = 1, 5$. The same set of parameters outlined in Section 5 are followed. Figure 11, 12 and 13 show the performance of SPATS, RSI and LATSI respectively, in the multi-agent setting. Our observations overall are similar to those in Section 5. The probabilistic decision-making nature of SPATS ensures that its performance multiplies by its number of agents. RSI relies on information theoretic decision making (no randomness).

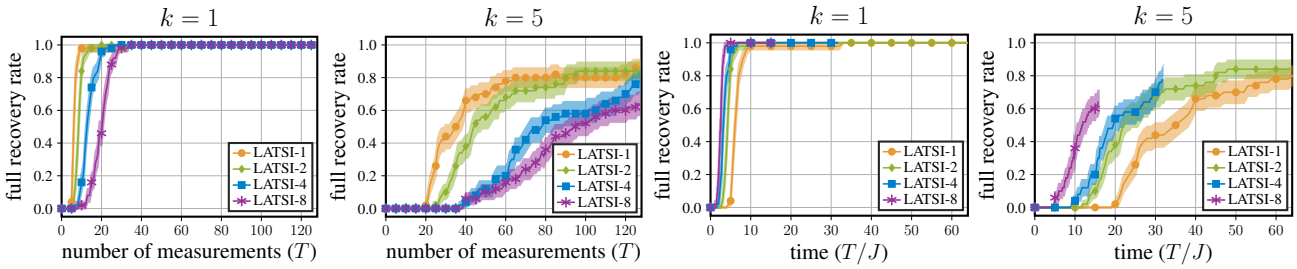


Figure 10: Full recovery rate of LATSI with 1, 2, 4 and 8 agents for $n = 128$, $k = 1, 5$

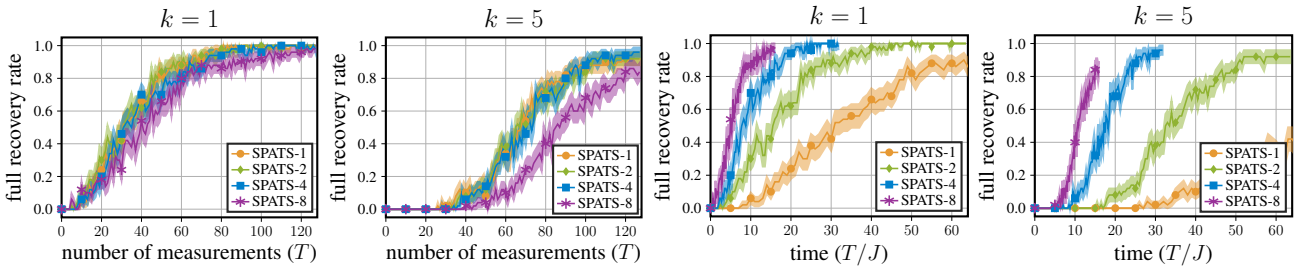


Figure 11: Full recovery rate of SPATS with 1, 2, 4 and 8 agents for $n = 16 \times 16$, $k = 1, 5$

Hence, this lack of randomness in its reward function together with the poor approximation of mutual information for $k > 1$, results in a poor performance generally worsening with more agents. The multi-agent performance of LATSI shows similar trends as SPATS given its partly probabilistic reward function. However, LATSI is not as efficient as SPATS given its poor approximation of mutual information due to the larger size of the environment. These results further support the analysis in Section 5.

1-DIMENSIONAL VS 2-DIMENSIONAL SEARCH SPACE

In Figure 14, we compare the performance of SPATS and LATSI for 1-dimensional (1d) and 2-dimensional (2d) search spaces with the same length $n = 128$. We compare a 1d search space of length $n = 128$ with a 2d search space of length $n = 8 \times 16$ (i.e. flattened length $n = 128$) for two sparsity rates $k = 1, 5$. The same set of parameters outlined in Section 5 are followed. As evident in this figure, while 1-dimensional SPATS and LATSI have a similar performance for $k = 1$, with larger k they perform better in a 2d space rather than in 1d. The reason for this behavior is our region sensing assumption. Specifically, region sensing in the 2-dimensional grid expands the available action space and allows for a larger feasible action set compared to the 1-dimensional grid. Mainly, the larger action set helps our algorithms to choose better sensing actions in the 2-dimensional space, particularly when multiple targets are present.

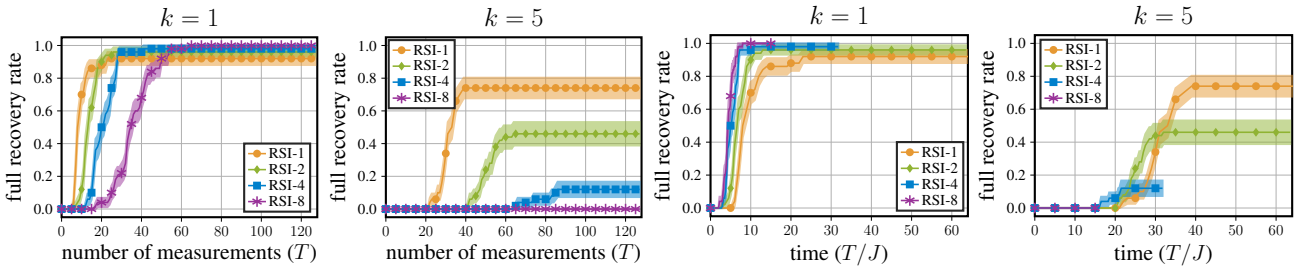


Figure 12: Full recovery rate of RSI with 1, 2, 4 and 8 agents for $n = 16 \times 16$, $k = 1, 5$

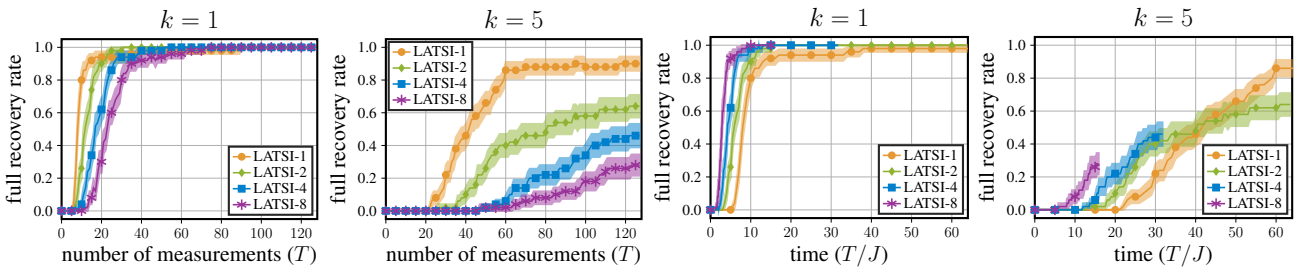
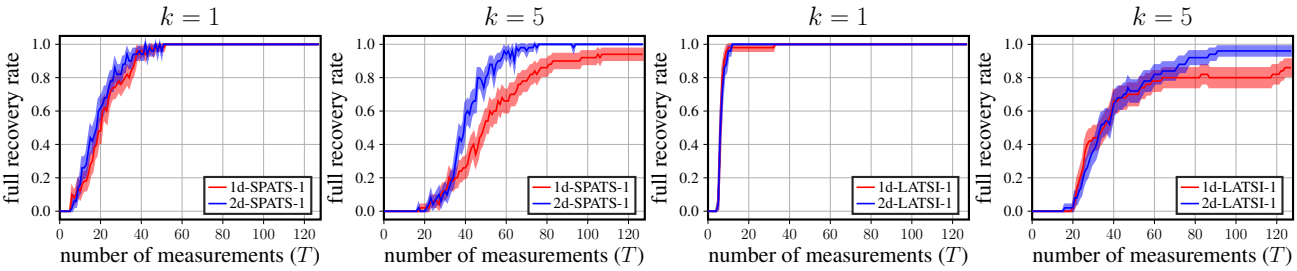
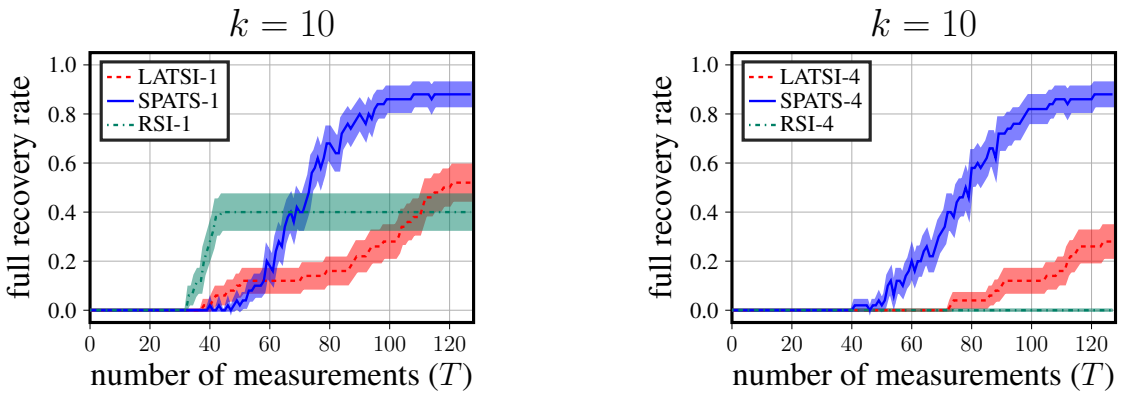


Figure 13: Full recovery rate of LATSI with 1, 2, 4 and 8 agents for $n = 16 \times 16$, $k = 1, 5$



(a) SPATS (b) LATSI

Figure 14: Full recovery rate in 1-dimensional ($n = 128$) vs 2-dimensional ($n = 8 \times 16$) search space with 1 agent for sparsity rates $k = 1, 5$



(a) single agent (b) four agents

Figure 15: Full recovery rate of SPATS, LATSI and RSI for 1 and 4 agents for sparsity rate $k = 10$

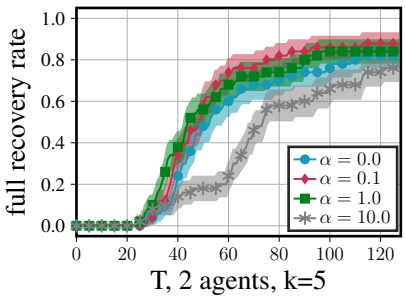


Figure 16: Effect of α in full recovery performance of LATSI with multiple agents and multiple targets

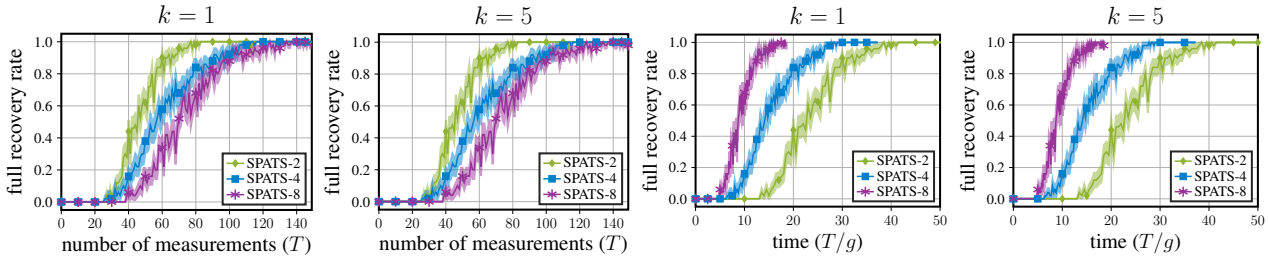


Figure 17: Full recovery rate of SPATS in the presence of communication failure with 2, 4 and 8 agents for sparsity rates $k = 1, 5$

SPARSITY RATE

In this section, we provide an additional set of results for sparsity rate of $k = 10$ in a 1-dimensional search space of length $n = 128$. Figure 15 illustrates the full recovery rate of SPATS, LATSI and RSI for one and four agents, respectively. Overall, we see that increasing the number of targets from $k = 5$ to $k = 10$ reduces the performance of all algorithms. Both figures confirm our previous observations that for sparsity rates of $k > 1$, SPATS algorithm outperforms LATSI and RSI. In the case of single agent, the poor performance of LATSI and RSI can be traced back to poor approximation of mutual information for $k > 1$ (see Section 5). In the four-agent scenario, RSI and LATSI are significantly worse than SPATS due to lack of randomness in information-greedy approaches (see Section 5 for details).

SENSITIVITY ANALYSIS FOR LATSI

Recall tuning parameter α as the scaling factor in (33) which is used to combine Laplace-TS and RSI's reward functions in LATSI's reward function $\boldsymbol{R}^+$. As evident by (33), increasing the value of α in $\boldsymbol{R}^+$ increases the weight of the reward computed by Laplace-TS. We experiment with different values of $\alpha \in \{0, 0.1, 1, 10\}$ to determine it's effect on the full recovery rate performance. Since it is our aim to exploit parallelization, we perform the simulations with 2 agents and $k = 5$ in a 1-dimensional search space of length $n = 128$. Plotting the results in Figure 16, we observe that the algorithm's performance is robust for a wide range of α. Based on this observation, we chose $\alpha = 1$ for all experiments with LATSI.

ROBUSTNESS TO UNRELIABLE INTER-AGENT COMMUNICATION

In this section, we provide experimental results showing the robustness of SPATS to unreliable communication among agents in asynchronous multi-agent active search. Using the same parameters and setting described in Section 5, we focus on a 2-dimensional search space where we estimate a k-sparse signal $\boldsymbol{\beta}$ with length $n = 8 \times 16$ and two sparsity rates of $k = 1, 5$. Assuming J number of agents are actively searching for the targets and asynchronously communicating their measurements to each other, we introduce unreliability in inter-agent communication by randomly dropping, for each agent, up to three of its last received new measurement messages from other agents. We measure the mean and standard error of the full recovery rate as a function of the number of measurements T as well as time over 50 random trials. We vary the number of agents J between 2, 4 and 8. Figure 17 shows that the agents are able to fully recover all targets even in this additionally introduced unreliable communication setup. The probabilistic nature of SPATS allows the agents to maintain their performance even when they cannot access all measurements from other agents.

References

Kyounghwa Bae and Bani K Mallick. Gene selection using a two-level hierarchical Bayesian model. *Bioinformatics*, 20 (18):3423–3430, 2004.

Thomas Blumensath and Mike E Davies. Iterative hard thresholding for compressed sensing. *Applied and computational harmonic analysis*, 27(3):265–274, 2009.

Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11), 2004.

Mário AT Figueiredo. Adaptive sparseness for supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 25(9):1150–1159, 2003.

Alan E Gelfand and Adrian FM Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, 85(410), 1990.

Kirthevasan Kandasamy, Akshay Krishnamurthy, Jeff Schneider, and Barnabás Póczos. Parallelised bayesian optimisation via thompson sampling. In *International Conference on Artificial Intelligence and Statistics*, 2018.

Yifei Ma, Roman Garnett, and Jeff Schneider. Active search for sparse signals with region sensing. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Arian Maleki. Approximate message passing algorithms for compressed sensing. *a degree of Doctor of Philosophy, Stanford University*, 2011.

Elaine Crespo Marques, Nilson Maciel, Lirida Naviner, Hao Cai, and Jun Yang. A review of sparse recovery algorithms. *IEEE Access*, 7:1300–1322, 2018.

Deanna Needell and Joel A Tropp. CoSaMP: iterative signal recovery from incomplete and inaccurate samples. *Communications of the ACM*, 53(12):93–100, 2010.

Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Asilomar*. IEEE, 1993.

Daniel Russo and Benjamin Van Roy. An information-theoretic analysis of thompson sampling. *Journal of Machine Learning Research (JMLR)*, 17(1), 2016.

Daniel Russo and Benjamin Van Roy. Learning to optimize via information-directed sampling. *Operations Research*, 66(1): 230–252, 2017.

Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. A tutorial on Thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, 2018.

Zhilin Zhang and Bhaskar D Rao. Sparse signal recovery with temporally correlated source vectors using sparse Bayesian learning. *IEEE Journal of Selected Topics in Signal Processing*, 5(5):912–926, 2011.