Bandit Theory meets Compressed Sensing for high dimensional Stochastic Linear Bandit

Alexandra Carpentier
Sequel team, INRIA Lille - Nord Europe

Remi Munos
Sequel team, INRIA Lille - Nord Europe

Abstract

We consider a linear stochastic bandit problem where the dimension $K$ of the unknown parameter $\theta$ is larger than the sampling budget $n$. In such cases, it is in general impossible to derive sub-linear regret bounds since usual linear bandit algorithms have a regret in $O(K\sqrt{n})$. In this paper we assume that $\theta$ is $S$-sparse, i.e. has at most $S$ non-zero components, and that the space of arms is the unit ball for the $\|\cdot\|_2$ norm. We combine ideas from Compressed Sensing and Bandit Theory and derive an algorithm with a regret bound in $O(S\sqrt{n})$. We detail an application to the problem of optimizing a function that depends on many variables but among which only a small number of them (initially unknown) are relevant.

Introduction

We consider a linear stochastic bandit problem in high dimension $K$. At each round $t$, from 1 to $n$, the player chooses $x_t$ in a fixed set of arms and receives a reward $r_t = \langle x_t, \theta + \eta_t \rangle$, where $\theta \in \mathbb{R}^K$ is an unknown parameter and $\eta_t$ is a noise term. Note that $r_t$ is a (noisy) projection of $\theta$ on $x_t$. The goal is to maximize the sum of rewards.

We are interested in cases where the number of rounds is much less than the dimension of the parameter, i.e. $n \ll K$. This is new in bandit literature but useful in practice, as illustrated by the problem of gradient ascent for a high-dimensional function, described later.

As $n \ll K$, it is in general impossible to even estimate $\theta$ in an accurate way. It is thus necessary to restrict the setting, and the assumption we consider here is that $\theta$ is sparse. We assume also that the set of arms to which $x_t$ belongs is the unit ball with respect to the $\|\cdot\|_2$ norm, induced by the inner product.

Bandit Theory meets Compressed Sensing

Our problem asks in an urgent way the fundamental question at the heart of bandit theory, namely the exploration versus exploitation dilemma. More precisely, when the dimension of the space $K$ is smaller than the budget $n$, it is possible to project the parameter $\theta$ at least once on each direction of a basis (e.g. the canonical basis): it is thus possible to explore efficiently. In our setting we assume that $K \gg n$ and it is thus not possible anymore to project even once on each direction of any basis of the space: we thus require a cleverer exploration technique.

Compressed Sensing provides us with ideas on how to explore, i.e. estimate $\theta$, provided that it is sparse, with few measurements: it is thus possible to roughly estimate its support without spending too much of the budget. The idea is to project $\theta$ on random (isotropic) directions such that each reward sample provides equal information about all coordinates of $\theta$. This is the reason why we emphasized the fact that the set of arm is the unit ball, as we need to be able to project $\theta$ in each direction of the space. Then using regularization (Hard Thresholding, Lasso, Dantzig selector...) enables to recover the support of the parameter. For some references on Compressed Sensing, see e.g. (Candes and Tao, 2007; Chen et al., 1999; Blumensath and Davies, 2009). Note however that such a technique allows only to retrieve a rough estimate of $\theta$ and is not designed for the purpose of maximizing the sum of rewards.

Bandit Theory is then a good tool to address this second issue, namely maximizing the sum of rewards by efficiently balancing between exploration and exploitation.

---

1 It is important to explore the space in order to build a good estimate of all components of $\theta$ in order to know which arms are the best ones.

2 It is important to exploit, i.e. to pull the empirical best arms in order to maximize the sum of rewards.
tion. In our setting, once a rough estimate of the (restricted) support of $\theta$ is available, we use a linear bandit algorithm. References on linear stochastic bandits include the works of Rusmevichientong and Tsitsiklis (2008); Dani et al. (2008); Filippi et al. (2010) and the recent work by Abbasi-yadkori et al. (2011).

Our contributions are the following.

- We provide an algorithm that mixes ideas of Compressed Sensing and Bandit Theory for solving the exposed problem. It has a regret\(^4\) which is of order $O(S\sqrt{n})$.
- We give a detailed example of an application of this setting to high dimensional gradient ascent when the gradient is sparse. We first explain why the setting of gradient ascent can be seen as a bandit problem. We then display numerical experiments supporting our belief that our algorithm provides an efficient way for solving the problem of high dimensional gradient ascent on functions that depend only on a small number of relevant variables.

A similar setting is also studied in the paper (Abbasi-yadkori et al., 2012), published simultaneously. Their assumption on the noise is however radically different: unlike in our setting they consider a noise to signal ratio that is different depending on whether the direction where they sample is flat or not. It is thus difficult to compare the results.

We formalize the setting in Section 1, and recall briefly what linear bandits can achieve when the dimension $K$ is low. We then describe the algorithm we propose for this problem, and give the main results in Section 2. We detail in Section 3 the application to gradient ascent and provide numerical experiments.

1 Setting and a useful existing result

1.1 Description of the problem

We consider a linear bandit problem in dimension $K$. An algorithm (or strategy) $\Alg$ is given a budget of $n$ pulls. At each round $1 \leq t \leq n$ it selects an arm $x_t$ in the set of arms $B_K$, which is the unit ball for the $\|\cdot\|_2$-norm induced by the inner product. It then receives a reward

$$r_t = \langle x_t, \theta + \eta_t \rangle,$$

where $\eta_t \in \mathbb{R}^K$ is an i.i.d. white noise\(^5\) that is independent from the past actions, i.e. from $\{(x_t')_{t' \leq t}\}$ and $\theta \in \mathbb{R}^K$ is an unknown parameter.

We define the performance of algorithm $\Alg$ as

$$L_n(\Alg) = \sum_{t=1}^{n} \langle \theta, x_t \rangle.$$  \hspace{1cm} (1)

Note that $L_n(\Alg)$ differs from the sum of rewards $\sum_{t=1}^{n} r_t$ but is close in high probability. For example if we assume that each $\eta_{k,t}$ is bounded by $\frac{1}{2}\sigma_k$, we know by Azuma inequality (because $x_t$ depends only of $(\eta_{k,s})_{s \leq t}$) that with probability $1 - \delta$, we have $\sum_{t=1}^{n} r_t = L_n(\Alg) + \sum_{t=1}^{n} (\eta_{k,t}, x_t) \leq L_n(\Alg) + \sqrt{2\log(1/\delta)}\|\theta\|_2\sqrt{n}$. Note that this result can be extended to sub-gaussian random variables $\eta_{k,t}$.

If the parameter $\theta$ was known, we could define an optimal fixed strategy $\Alg^*$ that always picks $x^* = \max_{x \in B_K} \langle \theta, x \rangle$ in order to maximize the performance. Here, $x^* = \frac{\theta}{\|\theta\|_2}$. The performance of $\Alg^*$ is given by

$$L_n(\Alg^*) = n\|\theta\|_2.$$  \hspace{1cm} (2)

We define the regret of an algorithm $\Alg$ with respect to this optimal strategy as

$$R_n(\Alg) = L_n(\Alg^*) - L_n(\Alg).$$  \hspace{1cm} (3)

We consider the class of algorithms that do not know the parameter $\theta$. Our objective is to find an adaptive strategy $\Alg$ that always picks $x^* = \max_{x \in B_K} \langle \theta, x \rangle$ in order to minimize the regret.

For a given $t$, we write $X_t = (x_1; \ldots; x_t)$ the matrix in $\mathbb{R}^{K \times t}$ of all chosen arms, and $R_t = (r_1, \ldots, r_t)^T$ the vector in $\mathbb{R}^t$ of all rewards, until time $t$.

In this paper, we consider the case where the dimension $K$ is much larger than the budget, i.e. $n \ll K$. As already mentioned, it is impossible in general to estimate accurately the parameter and thus achieve a sub-linear regret. This is the reason why we make the assumption that $\theta$ is $S$-sparse (i.e. there are at most $S$ components of $\theta$ which are not 0) with $S < n$.

1.2 A useful algorithm for Linear Bandits

In this paper we will use the algorithm $\text{Confidence Ball}_2$ that is described in the article of Dani et al. (2008), and which we abbreviate by $\text{CB}_2$. We recall here briefly the algorithm and the corresponding regret bound.

This algorithm is designed for stochastic linear bandit in dimension $d$, i.e. the bandit parameter $\theta$ is in $\mathbb{R}^d$, and $d$ is smaller than the budget $n$. This is the reason...
why we can not immediately apply this algorithm to the problem described in the previous subsection.

The pseudo-code of the algorithm is presented in Figure 1. The idea is to build an ellipsoid of confidence for the parameter $\theta$, namely $B_t = \{ \nu : ||\nu - \hat{\theta}_t||_{2,A} \leq \sqrt{t} \}$ where $||u||_{2,A} = u^T A u$ and $\hat{\theta}_{t+1} = A_{t+1}^{-1} X_t R_t$, and to pull the arm with largest inner product with a vector in $B_t$, i.e. the arm such that $x_t = \arg\max_{x \in B_t} \max_{\nu \in B_t} (\nu, x)$.

Note that this algorithm is intended for general shapes of the set of arms. We can thus apply it in the particular case where the set of arms is the unit ball for the $||.||_2$ norm in $\mathbb{R}^d$, i.e. $B_d$ and this case is simpler. At first, it is easier to find a span in the set of arms: we can just take the canonical basis of $\mathbb{R}^d$. Then we need to find the point of the confidence ellipsoid $B_t$ with largest norm in order to compute the upper confidence bound. Note also that we present here a simplified variant where the temporal horizon $n$ is known: the original version of the algorithm in (Dani et al., 2008) is anytime.

We recall here Theorem 2 of (Dani et al., 2008).

**Theorem 1 (ConfidenceBall$_2$)** Assume that the $\eta_t$ is an i.i.d. white noise, independent of the $(x_t)_{t \leq t}$ and that for all $k = \{1, \ldots, d\}$, $\exists \psi_k$ such that for all $t$, $|\eta_{t,k}| \leq \frac{1}{2} \psi_k$. If $n$ is large enough, we have with probability $1 - \delta$ the following bound for the regret of ConfidenceBall$_2(B_d, \delta)$:

$$R_n(A|g_{CB_2}) \leq 64d (||\theta||_2 + ||\sigma||_2) (\log(n^2/\delta))^{2\sqrt{n}}.$$  

2 The algorithm SL-UCB

Now we come back to our setting where $n \ll K$. We present here an algorithm, called Sparse Linear Upper Confidence Bound (SL-UCB).

2.1 Presentation of the algorithm

SL-UCB is divided in two main parts, (i) a first un-adaptive phase, that uses compressed sensing ideas and which is referred to as support exploration phase where we project $\theta$ on isotropic random vectors in order to select the arms that belong to what we call the active set $A$ and (ii) a phase that we call restricted linear bandit phase where we apply a linear bandit algorithm to the active set $A$ in order to balance exploration and exploitation and further minimize the regret. Note that the length of the support exploration phase is problem dependent.

This algorithm takes as parameters: $\sigma_2$ and $\bar{\theta}_2$ which are upper bounds respectively on $||\sigma||_2$ and $||\theta||_2$, and $\delta$ which is a (small) probability.

First, we define an exploring set as

$$E_{\text{exploring}} = \frac{1}{\sqrt{K}} \{-1, +1\}^K. \quad (4)$$

Note that $E_{\text{exploring}} \subset B_K$. We sample this set uniformly during the support exploration phase. This gives us some insight about the directions on which the parameter $\theta$ is sparse, using very simple concentration tools:

at the end of this phase, the algorithm selects a set of coordinates $A$, named active set, which

6Note that this idea is very similar to the one of compressed sensing.
are the directions where \( \theta \) is likely to be non-zero. The length of this phase is problem dependent and is built to be of order \( \frac{\sqrt{T}}{||\theta||_2} \): if the problem is difficult, i.e. \( ||\theta||_2 \) is small, we need a longer support exploration phase than if the problem is simple, i.e. \( ||\theta||_2 \) big. Note that the algorithm automatically adapts the length of this phase and that no lower bound on \( ||\theta||_2 \) is needed.

The Support Exploration Phase stops at the first time \( t \) such that (i) \( \max_k |\theta_{k,t}| - \frac{2b}{\sqrt{T}} > 0 \) for a well-defined \( b \) and (ii) \( t \geq \max_k |\theta_{k,t}| - \frac{\sqrt{\eta}}{\sigma_k} \).

We then exploit the information collected in the first phase, i.e. the active set \( \mathcal{A} \), by doing a linear bandit algorithm on the intersection of the unit ball \( B_K \) and the vector subspace spanned by the active set \( \mathcal{A} \), i.e. \( V_{cc}(\mathcal{A}) \). Here we choose to use the algorithm \( CB_2 \) described in (Dani et al., 2008). See Subsection 1.2 for an adaptation of this algorithm to our specific case: the set of arms is indeed the unit ball for the \( ||.||_2 \) norm in the vector subspace \( V_{cc}(\mathcal{A}) \).

The algorithm is described in Figure 2.

**Figure 2:** The pseudo-code of the SL-UCB algorithm.

Note that the algorithm computes \( \hat{\theta}_{k,t} \) using

\[
\hat{\theta}_{k,t} = \frac{K}{t} \left( \sum_{i=1}^{t} x_{k,i} r_i \right) = \left( \frac{K}{t} X_t R_t \right)_{k}. \tag{5}
\]

### 2.2 Main Result

We first state an assumption on the noise.

**Assumption 1** \((\eta_{k,t})_{k,t}\) is an i.i.d. white noise and \( \exists \sigma_k \) s.t. \( \eta_{k,t} \leq \frac{1}{2} \sigma_k \).

Note that this assumption is made for simplicity and that it could easily be generalized to, for instance, sub-Gaussian noise.

Under this assumption, we have the following bound on the regret.

**Theorem 2** Under Assumption 1, if we choose \( \sigma_2 \geq ||\sigma||_2 \) and \( \theta_2 \geq ||\theta||_2 \), the regret of SL-UCB is bounded with probability at least \( 1 - 5\delta \), as

\[
R_n(\mathcal{A}^\text{sl-ucb}) \leq 118(\theta_2 + \sigma_2)^2 \log(2K/\delta) S \sqrt{n}.
\]

The proof of this result is available in Section 4.

The algorithm SL-UCB uses at first an idea of compressed sensing: it explores by performing random projection and builds an estimate of \( \theta \). It then selects the support as soon as the uncertainty is small enough, and applies \( CB_2 \) to the selected support. The particularity of this algorithm is that the length of the support exploration phase adjusts to the difficulty of finding the support: the length of this phase is of order \( O\left( \frac{\sqrt{T}}{||\theta||_2} \right) \). More precisely, the smaller \( ||\theta||_2 \), the more difficult the problem is (as it is difficult to find the biggest components of the support), and the longer the Support Exploration Phase. But note that the regret does not deteriorate, as the smaller \( ||\theta||_2 \), the smaller also the loss at each step.

### 3 The gradient ascent as a bandit problem

The aim of this section is to propose a local optimization technique to maximize a function \( f : \mathbb{R}^K \to \mathbb{R} \) when the dimension of \( K \) is very high and when we can only sample \( n \) times this function with \( n \ll K \). We assume that the function \( f \) depends on a small number of relevant variables: it corresponds to the assumption that the gradient of \( f \) is sparse.

A well-known local optimization technique is gradient ascent, where one computes the gradient \( \nabla f(u) \) of \( f \) at point \( u \), move in the direction of the gradient, and then iterate \( n \) times. See for instance the book of Bertsimas (1999) for an exhaustive survey on gradient methods.

#### 3.1 Formalization

The objective is to apply gradient ascent to a differentiable function \( f \). Assume that we are allowed to do only \( n \) queries to the function. We call \( u_t \) the \( t \)-th point where we sample \( f \), and choose it such that \( ||u_{t+1} - u_t||_2 = \epsilon \), where \( \epsilon \) is the gradient step.

Note that by the theorem of intermediate values
where $w_t$ is an appropriate barycentre of $u_t$ and $u_{t-1}$.

We can thus model the problem of efficient gradient ascent by a linear bandit problem where the reward is the approximation error.

Remark on the linearity assumption: Matching the stochastic bandit model in Section 1 to the problem of gradient ascent corresponds to assuming that the function is (locally) linear in a neighborhood of $u_0$, and that we have in this neighborhood $f(u_{t+1}) - f(u_t) = \langle u_{t+1} - u_t, \nabla f(u_0) + \eta_{t+1} \rangle$, where the noise $\eta_{t+1}$ is i.i.d. This setting is very restrictive: we made it in order to offer a first, simple solution for the problem. When the function is not linear, there is an additional approximation error.

3.2 Numerical experiment

In order to illustrate the mechanism of our algorithms, we apply SL-UCB to a quadratic function in dimension 100 where only two dimensions are informative (we represent in Figure 4 the projection of the function in these two informative directions). Figure 4 shows the trajectory of the algorithm, projected in the subspace of dimension 2 where the function is not constant.

Note that at the beginning of the ascent, the projection of the steps on relevant directions are very small because we search for the good support and thus also move in other directions of the space than the subspace of dimension 2 where the gradient lies. However, the algorithm quickly concentrates on the good support of the gradient.

We now want to illustrate the performances of SL-UCB. We fix the number of pulls to 100, and we try different values of $K$, in order to have results for different values of $\frac{K}{n}$. The higher this quantity, the more difficult the problem. We choose a quadratic function varying in $S = 10$ directions$^8$.

We compare our algorithm SL-UCB with two strategies: the “oracle” gradient strategy (OGS), i.e. a gradient algorithm with access to the full gradient of the function$^9$, and what we call random best direction (BRD), that is to say a strategy that, at a given point, chooses a random direction, observes the value of the function a step further in this direction, and goes to that point if the value of the function at this point is higher.

We keep the same function when $K$ varies. It is the quadratic function $f(x) = \sum_{k=1}^{10} -20(x_k - 25)^2$.

Each of the 100 pulls corresponds to an access to the full gradient of the function at a chosen point.

\[
R_n(\text{Alg}) = \max_{x \in B_2(u_0, \alpha \epsilon)} f(x) - f(u_n),
\]

where $u_n$ is the terminal point of algorithm $\text{Alg}$. The regret is in $O(S\epsilon\sqrt{n})$ for SL-UCB.

Remark on the noise: Assumption 1, which states that the noise added to the function is of the form $\langle u_t - u_{t-1}, \eta_t \rangle$ is specially suitable for gradient ascent because it corresponds to the cases where the noise is an approximation error and depends on the gradient step.

Figure 4: Illustration of the trajectory of algorithm SL-UCB with a budget $n = 50$, with a zoom at the beginning of the trajectory to illustrate the support exploration phase. The levels of gray correspond to the contours of function.
point is better than what it was before. We report the difference between the value at the final point of the algorithm and the value at the beginning of the algorithm, i.e. the regret of the algorithm. The results are available in Figure 5.

<table>
<thead>
<tr>
<th>$K/n$</th>
<th>OGS</th>
<th>SL-UCB</th>
<th>BRD</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1.875 $10^9$</td>
<td>1.723 $10^9$</td>
<td>2.934 $10^4$</td>
</tr>
<tr>
<td>10</td>
<td>1.875 $10^9$</td>
<td>1.657 $10^9$</td>
<td>1.335 $10^4$</td>
</tr>
<tr>
<td>100</td>
<td>1.875 $10^9$</td>
<td>1.552 $10^9$</td>
<td>5.675 $10^3$</td>
</tr>
</tbody>
</table>

Figure 5: We report, for different values of $K/n$ and different strategies, the value of $f(u_n) - f(u_0)$.

Note that the performances of our algorithm is worse than the “oracle” gradient strategy. This is not surprising because SL-UCB is only given partial information on the gradient. However it performs much better than the random best direction. Note that the bigger $K/n$, the more impressive the improvements of SL-UCB over the random best direction strategy. This can be explained by the fact that the larger $K/n$, the less probable it is that the random direction strategy picks a direction of interest, whereas our algorithm is built for dealing with such problems.

### 4 Analysis of the SL-UCB algorithm

#### 4.1 Event $\xi$ of interest

**Step 0:** Bound on the variations of $\hat{\theta}_t$ around its mean during the Support Exploration Phase

Note that as $x_{k,t} = \frac{1}{\sqrt{K}}$ or $x_{k,t} = -\frac{1}{\sqrt{K}}$ during the Support Exploration Phase, the estimate $\hat{\theta}_t$ of $\theta$ during the Support Exploration Phase is such that, for any $t_0 \leq T$ and any $k$

$$\hat{\theta}_{k,t_0} = \frac{K}{t_0} \left( \sum_{t=1}^{t_0} x_{k,t} \theta_t \right) = \frac{K}{t_0} \left( \sum_{t=1}^{t_0} x_{k,t} \sum_{k' \neq k} x_{k',t} (\theta_{k'} + \eta_{k',t}) \right)$$

$$= \frac{K}{t_0} \sum_{t=1}^{t_0} x_{k,t}^2 \theta_k + \frac{K}{t_0} \sum_{t=1}^{t_0} x_{k,t} \sum_{k' \neq k} x_{k',t} \theta_{k'} + \frac{K}{t_0} \sum_{t=1}^{t_0} x_{k,t} \sum_{k' \neq k} x_{k',t} \eta_{k',t}$$

$$= \theta_k + \frac{1}{t_0} \sum_{t=1}^{t_0} \sum_{k' \neq k} b_{k,k',t} \theta_{k'} + \frac{1}{t_0} \sum_{t=1}^{t_0} \sum_{k' \neq k} b_{k,k',t} \eta_{k',t},$$

where $b_{k,k',t} = K x_{k,t} x_{k',t}$.

Note that as the $x_{k,t}$ are i.i.d. random variables such that $x_{k,t} = \frac{1}{\sqrt{K}}$ with probability $1/2$ and $x_{k,t} = -\frac{1}{\sqrt{K}}$ with probability $1/2$, the $(b_{k,k',t})_{k' \neq k}$ are i.i.d. Rademacher random variables, and $b_{k,k,t} = 1$.

**Step 1:** Study of the first term. Let us first study

$$\frac{1}{t_0} \sum_{t=1}^{t_0} \sum_{k' \neq k} b_{k,k',t} \theta_{k'}.$$  

Note that the $b_{k,k',t} \theta_{k'}$ are $(K - 1)T$ zero-mean independent random variables and that among them, $\forall k' \in \{1, \ldots, K\}$, $t_0$ of them are bounded by $\theta_{k'}$, i.e. the $(b_{k,k',t} \theta_{k'})$. By Hoeffding’s inequality, we thus have with probability $1 - \delta$ that

$$| \frac{1}{t_0} \sum_{t=1}^{t_0} \sum_{k' \neq k} b_{k,k',t} \theta_{k'} | \leq$$
Step 4: Arms that are not in $\mathcal{A}$ Let us consider an arm $k$ such that $|\theta_k| \geq \frac{3b\sqrt{2}}{n^{1/4}}$. Note that $T \geq T_{\min} = \frac{\beta^2}{\|\theta\|^2} \sqrt{n}$ on $\xi$. We thus know that on $\xi$

$$|\hat{\theta}_{k,T} - |\theta_k| - \frac{b}{\sqrt{T}} \geq \frac{3b\sqrt{2}||\theta||_{2}}{n^{1/4}} - \frac{b\sqrt{||\theta||_{2}}}{n^{1/4}} \geq \frac{2b}{\sqrt{T}}.$$ 

This means that $k \in \mathcal{A}$ on $\xi$. We thus know that $|\theta_k| \geq \frac{3b\sqrt{2}}{n^{1/4}}$ implies on $\xi$ that $k \in \mathcal{A}$.

Step 2: Arms that are not in $\mathcal{A}$ Now let us consider an arm $k$ such that $|\theta_k| < \frac{b}{\sqrt{T}}$. Then on $\xi$, we know that

$$|\hat{\theta}_{k,T} < |\theta_k| + \frac{b}{\sqrt{T}} \leq \frac{b}{2\sqrt{T}} + \frac{b}{2\sqrt{T}} \leq 2b\sqrt{\frac{n}{T}}.$$ 

This means that $k \in \mathcal{A}^c$ on $\xi$. This implies on $\xi$, if $|\theta_k| = 0$, then $k \in \mathcal{A}^c$.

Step 3: Summary Finally, we know that $\mathcal{A}$ is composed of all the $|\theta_k| \geq \frac{3b\sqrt{2}}{n^{1/4}}$, and that it contains only positives $\theta_k$, i.e. at most $S$ elements since $\theta$ is $S$-sparse. We write $A_{\min} = \{k : |\theta_k| \geq \frac{3b\sqrt{2}}{n^{1/4}}\}$

4.4 Comparison of the best element on $\mathcal{A}$ and on $\mathcal{B}$

Now let us compare $\max_{x \in \mathcal{A}^c} \{\mathbb{E}_k(\theta, x_t)\}$ and $\max_{x \in \mathcal{B}_k} (\theta, x_t)$.
At first, note that \( \max_{x_t \in B_K} (\theta, x_t) = ||\theta||_2 \) and that \( \max_{x_t \in V ec(A^c \cap B_K)} (\theta, x_t) = ||\theta^A||_2 = \sqrt{\sum_{k=1}^{K} \theta_k^2 1 \{ k \in A \}} \), where \( \theta_{A^c} = \theta_k \) if \( k \in A \) and \( \theta_{A^c} = 0 \) otherwise. This means that
\[
\max_{x_t \in B_K} (\theta, x_t) - \max_{x_t \in V ec(A^c \cap B_K)} (\theta, x_t) = ||\theta||_2 - ||\theta^A||_2 = \left| \left| \theta \right| \right|_2 - \left| \left| \theta^A \right| \right|_2 \leq \sum_{k \in A^c} \theta_k^2 \leq \sum_{k \in A^c} \theta_k^2 \leq \frac{9S\delta^2}{\sqrt{n}}.
\]

### 4.5 Expression of the regret of the algorithm

Assume that we launch algorithm \( ConfidenceBall_2(V ec(A) \cap B_K, \delta, T) \) at time \( T \) where \( A \subset Supp(\theta) \) with a budget of \( n_1 = n - T \) samples. In paper (Dani et al., 2008), they prove that on an event \( \xi_2(V ec(A) \cap B_K, \delta, T) \) of probability \( 1 - \delta \) the regret of algorithm \( ConfidenceBall_2 \) is bounded by
\[
R_n(AlgCB_2(V ec(A) \cap B_K, \delta, T)) \leq 64|A| \left( ||\theta||_2 + ||\sigma||_2 \right) (\log(n^2/\delta)^2 \sqrt{n}).
\]

Note now that as \( A \subset Supp(\theta) \), there is \( \xi_2(V ec(A) \cap B_K, \delta, T) \subset \xi_2(V ec(Supp(\theta)) \cap B_K, \delta, T) \) (see paper (Dani et al., 2008) for more details on the event \( \xi_2 \)). We thus now that, conditional to \( T \), with probability \( 1 - \delta \), the regret is bounded for any \( A \subset Supp(\theta) \) as \( R_n(AlgCB_2(V ec(A) \cap B_K, \delta, T)) \leq 64S \left( ||\theta||_2 + ||\sigma||_2 \right) (\log(n^2/\delta)^2 \sqrt{n}).\)

By doing an union bound on all possible values for \( T \) (i.e. from 1 to \( n \)), we obtain that on an event \( \xi_2 \) whose probability is higher than \( 1 - \delta \), \( R_n(AlgCB_2(V ec(A) \cap B_K, \delta, T)) \) is bounded by \( 64S \left( ||\theta||_2 + ||\sigma||_2 \right) (\log(n^2/\delta)^2 \sqrt{n}).\)

We thus have on \( \xi \cup \xi_2 \), i.e. with probability higher than \( 1 - 2\delta \) that
\[
R_n(AlgSL-UCB, \delta) \leq 2T \max_{t} ||\theta||_2 + \max_{t} R_n(AlgCB_2(V ec(A) \cap B_K, \delta, T)) + n \max_{x \in B_K} \langle x, \theta \rangle - \max_{x \in B_K \cap V ec(A_{max})} \langle x, \theta \rangle.
\]

By using this Equation, the maximal length of the support exploration phase \( T_{max} \) deduced in Step 2 of Subsection 4.2, and Equation 12, we obtain on \( \xi \)
\[
R_n \leq 64S \left( ||\theta||_2 + ||\sigma||_2 \right) (\log(n^2/\delta)^2 \sqrt{n}) + 18S\delta^2 \sqrt{n} + 9S\delta^2 \sqrt{n} \leq 118(\hat{\theta}_2 + \hat{\sigma}_2)^2 \log(2K/\delta)S \sqrt{n}.
\]

by using \( b = (\hat{\theta}_2 + \hat{\sigma}_2)^2 \sqrt{2 \log(2K/\delta)} \) for the third step.

**Conclusion**

In the paper we provided an algorithm for sparse linear bandits in high dimension. It has been designed using ideas from compressed sensing and bandit theory. Compressed sensing is used in the support exploration phase, in order to select the support of the parameter. A linear bandit algorithm from (Dani et al., 2008) is then applied to the small dimensional subspace defined by the first phase. The algorithm SL-UCB provides a regret of order \( O(S \sqrt{n}) \). Note that all the bound scales with the sparsity \( S \) of the unknown parameter \( \theta \) instead of the dimension \( K \) of the parameter (as is usually the case in linear bandits). We then provided an example of application for this setting, the optimization of a function in high dimension. There are to our minds two main directions for further researches.

- It could be interesting to deal with the case when the support of \( \theta \) evolves in time: it would be nice to have some assumptions on the way it has to change so that we are able to achieve sub linear regret. One idea would be to use techniques developed for adversarial bandits (see (Abernethy et al., 2008; Bartlett et al., 2008; Cesa-Bianchi and Lugosi, 2009; Koolen et al., 2010; Audibert et al., 2011), but also (Flaxman et al., 2005) for a more gradient-specific modeling) or also from restless/switching bandits (see e.g. (Whittle, 1988; Nino-Mora, 2001; Slivkins and Upfal, 2008; A. Garivier, 2011) and many others). This would be particularly interesting to model gradient ascent on e.g. convex function where the support of the gradient changes in space.

- The bound that we obtain is in \( O(S \sqrt{n}) \), and it would be interesting to see if it is possible to build an algorithm that has a regret in \( O(\sqrt{Sn}) \), which is a theoretic lower bound for this problem. Note that when an upper bound \( S' \) on the sparsity is available, it seems possible to obtain such a regret by replacing condition (ii) in the algorithm by \( t < \left( \frac{\hat{\theta}_2 + \hat{\sigma}_2}{S' \sqrt{n}} \right) \) for the Exploitation phase the algorithm in (Rusmevichientong and Tsitsiklis, 2008). The regret of such an algorithm would be in \( O(\sqrt{S'n}) \). But it is not clear whether it is possible to obtain such results when no upper bound on \( S \) is available.

**Acknowledgements**

This research was partially supported by Region Nord-Pas-de-Calais Regional Council, French ANR EXPLORA (ANR-08-COSI-004), the European Communitys Seventh Framework Programme (FP7/2007-2013) under grant agreement 231495 (project CompLACS), and by Pascal-2.
References


