Online and Bandit Algorithms Beyond $\ell_p$ Norms

Thomas Kesselheim* Marco Molinaro† Sahil Singla‡

October 26, 2022

Abstract

Vector norms play a fundamental role in computer science and optimization, so there is an ongoing effort to generalize existing algorithms to settings beyond $\ell_\infty$ and $\ell_p$ norms. We show that many online and bandit applications for general norms admit good algorithms as long as the norm can be approximated by a function that is “gradient-stable”, a notion that we introduce. Roughly it says that the gradient of the function should not drastically decrease (multiplicatively) in any component as we increase the input vector. We prove that several families of norms, including all monotone symmetric norms, admit a gradient-stable approximation, giving us the first online and bandit algorithms for these norm families.

In particular, our notion of gradient-stability gives $O(\log^2(\text{dimension}))$-competitive algorithms for the symmetric norm generalizations of Online Generalized Load Balancing and Bandits with Knapsacks. Our techniques extend to applications beyond symmetric norms as well, e.g., to Online Vector Scheduling and to Online Generalized Assignment with Convex Costs. Some key properties underlying our applications that are implied by gradient-stable approximations are a “smooth game inequality” and an approximate converse to Jensen’s inequality.

* (thomas.kesselheim@uni-bonn.de) Institute of Computer Science, University of Bonn.
† (mmolinaro@microsoft.com) Microsoft Research and PUC-Rio.
‡ (ssingla@gatech.edu) School of Computer Science, Georgia Tech.
## Contents

1 Introduction
1.1 Summary of Our Results ......................................................... 2
1.2 Results for Online Algorithms .................................................. 2
1.3 Results for Bandit Algorithms .................................................. 4
1.4 Gradient-Stable Approximations and their Key Properties ................. 6
1.5 Constructing Gradient-Stable Approximations of Norms ..................... 8

2 Gradient-Stability for Symmetric Norms and Compositions ................. 9
2.1 Top-k Norms .............................................................................. 9
2.2 Composition of Norms .............................................................. 13
2.3 Symmetric Norms ................................................................. 14

3 Gradient-Stability implies a Smooth Game Inequality ...................... 16

4 Applications to Online Algorithms .............................................. 18
4.1 Online Generalized Load Balancing ............................................ 18
4.2 Online Vector Scheduling ....................................................... 19
4.3 Online Generalized Assignment with Convex Costs ......................... 21

5 Applications to Bandit Algorithms ............................................. 23
5.1 Approximate Converse to Jensen’s Inequality ................................ 23
5.2 Bandits with Knapsacks ......................................................... 23
5.3 Bandits with Vector Costs .................................................... 27

6 Conclusions and Further Directions ........................................... 28

A Further Related Work and Missing Proofs .................................... 29
A.1 Further Related Work ............................................................ 29
A.2 Softmax and $\ell_p$ Norms ..................................................... 30
A.3 Smoothing of Top-k Norm in Polynomial Time ............................... 31
A.4 Composition of Norms ......................................................... 35
A.5 Explicit Representation of a Symmetric Norm in Polynomial Time ....... 38
1 Introduction

Many fundamental problems in optimization and computer science involve norms of vectors, implicitly or explicitly. A classic example is Load Balancing or machine scheduling, where we need to assign jobs to machines in order to minimize the maximum load on a machine. This objective is nothing but the $\ell_\infty$-norm of the vector of loads incurred by each machine. Other examples that explicitly deal with norms include Nearest Neighbors Problem where the distance between points is a norm and Discrepancy Minimization where we minimize a norm of the signed sum of vectors. There are many other applications that do not explicitly involve a norm but can be phrased as such. For instance, observe that every set of non-negative vectors $C$ defines a norm: $\|x\|_C := \sup_{c \in C} \sum_i c_i |x_i|$. An important example of this observation is that the class of XOS functions\(^1\), which contains submodular functions and has applications in mechanism design and submodular optimization [FV06, FGL15, CZ17, AKS21], can be viewed as norms.

For some more concrete examples of norms and its applications, we have the classic $\ell_p$-norms (including $\ell_\infty$) $\|x\|_p := (\sum_i |x_i|^p)^{1/p}$. Another example is the Top-$k$-norm, which is the sum of the largest $k$ absolute values of the coordinates of the vector; these provide another important family of interpolations between the $\ell_1$ ($k=d$) and $\ell_\infty$ ($k=1$) norms. More broadly we have the ordered norms, namely those of the form $\|x\|_\lambda := \sum_i w_i |x_i|^\lambda$ for non-negative weights $w_1 \geq w_2 \geq \ldots \geq w_d \in \mathbb{R}_+$, where $|x|^\lambda$ is the vector with the absolute value of the coordinates of $x$ sorted in non-increasing order; see the books [NP06, LNdG15] for their applications in location theory, and [BSS18, CS18] for their applications in $k$-clustering. Another rich subset of norms are Orlicz norms, which are defined as $\|x\|_f := \inf \{ \lambda > 0 \mid \sum_i f(\frac{|x_i|}{\lambda}) \leq 1 \}$ for any convex function $f$ with $f(0) = 0$. E.g., the $\ell_p$-norm is the special case when $f(z) = z^p$. See [ALS+18, SWY+19] for recent applications of these norms to regression.

We note that all the norms mentioned in the previous paragraph are monotone, namely $\|u\| \geq \|v\|$ whenever $u \geq v \geq 0$, and symmetric, namely they are invariant to the permutation of the coordinates of the space. Indeed, most “naturally occurring” norms satisfy these properties. See [CS19a] for further examples and [Bha13, Chapter IV] for basic properties of monotone symmetric norms.

Since norms play an important role in many areas, there have been several recent works trying to understand more general norms in different settings. To mention a few examples, there are works on offline Load Balancing and $k$-clustering with symmetric monotone norms [CS19a, CS19b, IS21], stochastic Load Balancing and spanning trees (also with symmetric monotone norms) [IS20, IS21], nearest-neighbor search [ANN+17, ANN+18] (first for symmetric monotone norms, and then for general norms), linear regression with Orlicz norms [ALS+18, SWY+19], and mean estimation with statistical queries [LNRW19]. However, all these works are either in an offline or a stochastic setting. Despite of all this progress, we are not aware of general techniques to obtain similar results for online (adversarial) problems. For example, the results from [CS19a, CS19b, IS21] on offline Load Balancing and related problems (the closest to the problems we consider) are obtained by rounding the solution of fairly complex convex programs, and it is unclear how to solve and round them online. This motivates our main question regarding online algorithms and online learning:

_How to design algorithms for online problems with general monotone norms?_

\(^1\)A functions $f : 2^V \to \mathbb{R}$ over a finite ground set $V$ is XOS iff it can be written as $f(S) := \max_{c \in C} \sum_{e \in S} c(e)$ for some set $C$ of non-negative weight vectors.
1.1 Summary of Our Results

In this paper we introduce the notion of gradient-stable approximations of norms (Definition 1.7) and use it to obtain new results in a unified manner for several classic online problems involving norms, including Online Generalized Load Balancing and Bandits with Knapsacks. The idea is to approximate the norm by a function that has the property that its gradient at \( x \in \mathbb{R}^d_+ \) does not to decrease drastically in any component when increasing \( x \) by a small amount. We show that if an approximation with this gradient-stable property has a multiplicative error \( \alpha \) and an additive error \( \gamma \) then we can obtain online algorithms with competitive ratio \( O(\alpha + \gamma) \).

Crucially, one of our key technical contributions is to show that every monotone symmetric norm admits a gradient-stable approximation with a multiplicative error \( \alpha = O(\log d) \) and an additive error \( \gamma = O(\log^2 d) \). This gives us \( O(\log^2 d) \)-approximations with monotone symmetric norms, the first such general results for the problems considered. We also prove that gradient-stability works well under norm compositions, which allow us to extend the results beyond symmetric norms.

1.2 Results for Online Algorithms

Online Generalized Load Balancing. In the fundamental problem of scheduling on unrelated machines to minimize makespan (see the books [BEY05, BN09]), the goal is to minimize the \( \ell_\infty \)-norm of the vector of all machine loads. The more general case where the norm is the \( \ell_p \)-norm \( \|x\|_p := (\sum_i x_i^p)^{1/p} \) has also been studied since the 70’s [CW75, CC76], since in many applications they better capture how well-balanced an allocation is [AAG+95]. These are some of the special cases of the Online Generalized Load Balancing problem [Mol17].

In Online Generalized Load Balancing there are \( m \) machines, and \( T \) jobs come one-by-one. Each job can be processed by the machines in \( k \) different ways, so the \( t \)-th job has a \( m \times k \) non-negative matrix \( C^{(t)} \) whose column \( j \) gives the loads \( (C^{(t)}_{1j}, C^{(t)}_{2j}, \ldots, C^{(t)}_{mj}) \) that the machines incur if the job is processed with option \( j \). When the \( t \)-th job arrives, the algorithm needs to select a processing option for it (namely a vector \( x^{(t)} \in \{0, 1\}^k \) with exactly one 1) based only on the jobs seen thus far. To measure the quality of the solution, it is given a norm \( \| \cdot \| \) over \( \mathbb{R}^m \). The goal is to minimize the norm of the total load incurred on the machines, namely \( \| \sum_{t=1}^T C^{(t)} x^{(t)} \| \). The performance of the algorithm is compared against the offline optimal solution \( \text{OPT} := \min_{x^*_1, \ldots, x^*_T} \| \sum_{t=1}^T C^{(t)} x^{(t)}_t \| \), and it is said to be a \( \beta \)-approximation if its total load is at most \( \beta \text{OPT} \).

For the classic setting of unrelated scheduling with makespan minimization (\( \ell_\infty \)-norm, diagonal matrices \( C^{(t)} \)), traditional algorithms give an \( O(\log m) \)-approximation [ANR95, AAF+97], and this was generalized for the setting of \( \ell_p \)-norms by [AAG+95, Car08] to obtain an \( O\left(\min\{p, \log m\}\right) \)-approximation for all \( p \in [1, \infty] \) (all these results are optimal, up to constant factors). For the problem of online routing (\( \ell_\infty \)-norm, but the columns of \( C^{(t)} \) form paths on a graph) [AAF+97], there is also an optimal \( O(\log m) \)-approximation. The results in [Mol17] give a \( O\left(\min\{p, \log m\}\right) \)-approximation for the problem with \( \ell_p \)-norm (and arbitrary matrices \( C^{(t)} \), thus generalizing the above). Optimal results are also known for specifically structured norms, in which case the problem is known as Online Vector Scheduling (more on this below). For more general norms, only the offline version of the problem was recently solved: [CS19a, CS19b, IS21] obtained constant-factor approximations for every monotone symmetric norm.
Our main result for this problem is the following (see Definition 1.7 for gradient-stability):

**Theorem 1.1** (Load Balancing). For Online Generalized Load Balancing, if the norm $\| \cdot \|$ admits a $\frac{1}{4}$-gradient-stable approximation with error $(\alpha, \gamma)$, where $\alpha$ is the multiplicative error and $\gamma$ is the additive error, then a greedy algorithm obtains an $O(\alpha + \gamma)$ competitive ratio. Moreover, this greedy algorithm is efficient given value and gradient oracle access to the $\frac{1}{4}$-gradient-stable approximation.

In the appendix (Lemma A.2), we show that a known approximation of $\ell_p$-norms for $p \in [1, \infty]$ is indeed a $\frac{1}{4}$-gradient-stable approximation with error $(1, \min\{p-1, \log m\})$. So, Theorem 1.1 directly recovers the optimal $O(\min\{p, \log m\})$-approximations from [AAG+95, Car08, Mol17]. Using instead our new approximation of symmetric norms (stated in Theorem 2.1) allows us to obtain the following polylog approximation for any monotone symmetric norm.

**Theorem 1.2.** Consider the Online Generalized Load Balancing problem with any monotone symmetric norm $\| \cdot \|$. Then, there is an algorithm that obtain an $O(\log m)$ competitive ratio. Moreover, this algorithm is efficient given Ball-Optimization oracle access to the norm $\| \cdot \|$.

We note that previously no polylog competitive algorithms were known even for Top-$k$ norm or ordered norms. Since even for the $\ell_\infty$-norm there is an $\Omega(\log m)$ lower bound on the approximation factor [ANR95], Theorem 1.2 is optimal up to the exact polylog.

**Applications Beyond Symmetric Norms.** Our insights (especially the general Theorem 1.1) have implications beyond symmetric norms. As one example, we consider the Online Vector Scheduling problem. In brief, this is a problem similar to Online Load Balancing but where each machine has $r$ resources (e.g., a server with CPU, network, and disk); when a job is assigned to a machine, it loads these resources in different ways. For each resource $i$, there is an “inner” norm that measures the distribution of the load on this resource across the machines. The algorithm’s goal is to minimize the largest (makespan) of these inner norms. (See Section 4.2 for a formal definition of the problem.)

Following up on previous works, [IKKP15] gave an optimal $O(\log m + \log r)$-approximation algorithm when the inner norms are $\ell_p$’s. We are able to go beyond their results and obtain a polylog approximation for Online Vector Scheduling with any monotone symmetric inner norms. Although it is not difficult to cast Online Vector Scheduling as a special case of the Online Generalized Load Balancing with a “nested norm”, this norm is not symmetric even if the inner norms are. Nonetheless, we show that in this case we can still obtain low-error gradient-stable approximations, which yields the following result.

**Theorem 1.3.** Consider the Online Vector Scheduling problem where the inner norms are arbitrary monotone symmetric norms. Then, there is an $O(\log^2 m + \log r)$-competitive algorithm. Moreover, this algorithm is efficient given Ball-Optimization oracle access to the inner norms.

We remark that in the special case where the inner norms are $\ell_p$, our result improves to an $O(\log m + \log r)$-approximation, which recovers the result from [IKKP15].

---

2 We use the definition in [CS19a] whereby Ball-Optimization oracle allows us to compute $\max_{v: \|v\| \leq 1} \langle x, v \rangle$ for any vector $x \in \mathbb{R}^d$ with a single oracle call.
Another related application is the Online k-Sided Placement problem \cite{KMRT15}; for simplicity we briefly describe the case \(k = 2\). There are \(k = 2\) sets of machines offering different services (e.g., storage and processing). Each machine has a capacity. Jobs come online and need to be assigned to one storage and one processing machine. When assigned to a machine pair, a job adds to the load of these machines and also incurs a cost, both load and cost depending on the job and machine pair. In its minimization version, the goal is to find an assignment minimizing the total cost while respecting the machine capacities. In the offline case, \cite{KMRT15} gives an algorithm that obtains optimal cost while violating the capacities by a factor of at most \((k + 1)\) (such violation is required for any non-trivial approximation in the cost). While the paper also consider an online maximization version of the problem, nothing is known for this minimization version.

As in Online Vector Scheduling (plus using a guessing of OPT) one can see that this problem is a special case of Online Load Balancing with a non-symmetric nested norm. Then using the same ideas, we can obtain an online algorithm that has cost at most \(O(\log m)\) times OPT and violates the capacities by at most a factor of \(O(\log m)\), where \(m\) is again the total number of machines.

As another application of our techniques, we consider a problem where the objective is not even a norm. In Online Generalized Assignment with Convex Costs (OnGAP), we again have \(m\) machines and \(T\) online jobs with \(k\) processing options each. However, now each job has both a cost and a duration that it will incur over the machines. There are two monotone convex functions \(f_{\text{cost}}, f_{\text{dur}} : \mathbb{R}_+^m \to \mathbb{R}_+\) that measure the total cost and total duration that a schedule induces on the machines. The goal is to find a schedule of jobs that minimizes the sum \(f_{\text{cost}} + f_{\text{dur}}\). This is an online and a generalized version of the classic Generalized Assignment Problem of Shmoys-Tardos \cite{ST93}, which is a building block of algorithms on a myriad of applications (for a sample see \cite{BR01, CK05, GKNS18, Mol19} and the book \cite{Gon07}).

In \cite{GKP13} the authors consider a special case of OnGAP where \(f_{\text{cost}}\) is the \(\ell_p\)-norm raised to the power of \(p\) and \(f_{\text{dur}}\) is the \(\ell_1\)-norm. They were motivated by applications to energy efficient scheduling and routing. In this special case they obtain an optimal \(O(p^p)\)-approximation. Here we can generalize their result to arbitrary symmetric \(f_{\text{cost}}\)’s and \(f_{\text{dur}}\)’s with a competitive ratio that depends on “growth order”, a standard dependence when working with non-homogeneous functions \cite{BGMS11, HK15, ABC16, Mol21}. Formally, a function \(f\) has growth order at most \(p\) if \(f(\alpha x) \leq \alpha^p f(x)\) for all \(\alpha \geq 1\) and \(x \geq 0\). E.g., the function \(\| \cdot \|_p\) (used in previous work), and more generally polynomials of degree \(p\) with non-negative coefficients, have growth order at most \(p\).

**Theorem 1.4.** Consider the problem OnGAP with functions \(f_{\text{cost}}, f_{\text{dur}}\) that are convex, monotone, and symmetric. Then there is an \(O(\log^{2p} m)\)-competitive algorithm for this problem, where \(p\) is the maximum of the growth order of \(f_{\text{cost}}\) and \(f_{\text{dur}}\). Moreover, this algorithm is efficient given Ball-Optimization access to the norms \(\| \cdot \|_{f_{\text{cost}}}\) and \(\| \cdot \|_{f_{\text{dur}}}\), where for a convex function \(f\) its norm is \(\| x \|_f := \inf \{ \lambda > 0 \mid f(\frac{x}{\lambda}) \leq 1\}\).

The lower bound from Online Load Balancing in \(\ell_\infty\)-norm \cite{ANR95} implies an \(\Omega(\log^p m)\) lower bound for this problem, so again our result is tight up to a factor of two in the exponent.

### 1.3 Results for Bandit Algorithms

Next we discuss our results for online learning problems with bandit (partial) feedback.
Bandits with Knapsacks. In this problem we are given a budget \( B \geq 0 \), a norm \( \| \cdot \| \), and a set of \( n \) actions. In each time step \( t \in [T] \), we take one of these actions \( x(t) \in \{ e_1, \ldots, e_n \} \), and then receive a scalar reward \( \langle r(t), x(t) \rangle \) and incur a \( d \)-dim vector cost \( C(t) x(t) \), where \( r(t) \in [0,1]^{1 \times n} \) is a row vector of rewards and \( C(t) \in [0,1]^{d \times n} \) is a cost matrix. Moreover, we also receive a bandit feedback \( C(t) x(t) \) and \( \langle r(t), x(t) \rangle \), i.e., we do not get to see the entire cost matrix or reward vector but only the part pertaining to the played action after having made the choice. We assume there is a null action in the action set that gives 0 reward and 0 vector cost, which allows us to skip some time steps. The goal is to maximize the total reward received while ensuring that the \( \| \cdot \| \) norm of the total cost vector is less than \( B \). After exhausting the budget, we are only allowed to choose the null action, and thus we obtain no further reward. The benchmark for BWK is any fixed fractional selection \( x^* \in \Delta_n \) of actions, where \( \Delta_n \) is the \( n - 1 \) dimensional simplex \( \{ x \in [0,1]^n \mid \sum_i x_i \leq 1 \} \).

This problem was first introduced in [BKS18] in the special case where the rewards and cost vectors are drawn i.i.d. and the norm is \( \ell_\infty \). Since then the problem has been generalized to adversarial rewards/costs [ISSS19] and the norm has been generalized to \( \ell_p \)-norms [KS20], obtaining tight \( O(\log d) \)-approximations when the optimal value is known. These works were motivated by applications such dynamic item pricing, repeated auctions, and dynamic procurement where the bandit actions consume a budget besides giving a reward. Our result vastly generalizes the previous settings to arbitrary norms that admit gradient-stable approximations.

**Theorem 1.5.** Consider the Bandits with Knapsacks problem for adversarial arrivals with a norm \( \| \cdot \| \). Suppose \( \| \cdot \| \) admits a \( \frac{1}{d} \)-gradient-stable approximation with error \( (\alpha, \gamma) \). Furthermore, let \( B \geq 4 \cdot (\alpha + \gamma) \cdot \|1\| \). Then, there exists an algorithm that takes \( \text{OPT}_{\text{BWK}} \) as its input and obtains reward at least

\[
\Omega\left(\frac{1}{\alpha + \gamma} \cdot \text{OPT}_{\text{BWK}}\right) - O\left(\frac{\text{OPT}_{\text{BWK}} \cdot \|1\|}{(\alpha + \gamma) \cdot B}\right) \cdot \text{REGRET},
\]

with probability \( 1 - p \), where \( \text{REGRET} = O(\sqrt{T n \log(n/p)}) \) and \( p \in [0,1] \) is a parameter. Moreover, this algorithm is efficient given gradient oracle access to \( \frac{1}{d} \)-gradient stable approximation of \( \| \cdot \| \).

In particular, for monotone symmetric norms \( \| \cdot \| \) this gives the guarantee

\[
\Omega\left(\frac{1}{\log^2 d} \cdot \text{OPT}_{\text{BWK}}\right) - O\left(\frac{\text{OPT}_{\text{BWK}} \cdot \|1\|}{\log^2 d \cdot B}\right) \cdot \text{REGRET},
\]

which generalizes the previous results while still obtaining a polylog \( d \) approximation. The assumption that \( \text{OPT}_{\text{BWK}} \) is known can be removed at a further multiplicative loss of \( \Theta(\log T) \), which is known to be unavoidable [ISSS19].

Our techniques to prove this theorem using gradient-stable norm approximations also apply to other bandit problems, e.g, the following “Bandits with Vector Costs” problem.

Bandits with Vector Costs. This problem is a natural generalization of the classic adversarial bandits problem [ACFS02] where we incur vector costs instead of scalar costs. The goal is to minimize a given norm of the total cost vector. Formally, we are given a set of \( n \) actions and in each time step \( t \) we take one of these actions \( x(t) \in \{ e_1, \ldots, e_n \} \). After taking the action we incur a vector cost \( C(t) x(t) \), where \( C(t) \in [0,1]^{d \times n} \) is a cost matrix, and receive a bandit feedback \( C(t) x(t) \), i.e., only for the played action. The goal of the algorithm is to minimize a given norm \( \| \cdot \| \) of its total cost vector compared to that of a fixed fractional selection \( x^* \in \Delta_n \) over the actions.
In the special case of \( \ell_p \)-norms, this problem was introduced in [KS20]. They obtained a tight \( O(\min\{p, \log d\}) \) approximation in this setting. Our work generalizes this result to all norms admitting gradient-stable approximations.

**Theorem 1.6.** Consider the problem Bandits with Vector Costs with a norm \( \| \cdot \| \). If \( \| \cdot \| \) admits a \( \frac{1}{4} \)-gradient-stable approximations with error \((\alpha, \gamma)\), then there exists an algorithm that guarantees 

\[
\sum_{t=1}^T C(t) \cdot x(t) \leq O\left( (\alpha + \gamma) \cdot \sum_{t=1}^T C(t) \cdot x^* + \alpha \cdot \|1\| \cdot \text{Regret} \right)
\]

with probability \( 1 - p \), where \( \text{Regret} = O\left( \sqrt{Tn \log(n/p)} \right) \) and \( p \in [0, 1] \) is a parameter. Moreover, this algorithm is efficient given value and gradient oracle access to this \( \frac{1}{4} \)-gradient stable approximation to \( \| \cdot \| \).

Again, in the case of monotone symmetric norms \( \| \cdot \| \) this gives the guarantee \( O(\log^2 d \cdot \|\text{OPT}\| + \log d \cdot \|1\| \cdot \text{Regret}) \), generalizing the result of [KS20] while still obtaining a polylog \( d \) approximation.

### 1.4 Gradient-Stable Approximations and their Key Properties

As mentioned above, the central concept underlying our algorithms is that of approximating norms with functions whose gradients are stable. Namely, we introduce the following definition.

**Definition 1.7 (Gradient-Stable Approximation).** We say that a norm \( \| \cdot \| \) admits a \( \delta \)-gradient-stable approximation with error \((\alpha, \gamma)\) if for every \( \epsilon > 0 \) there is a monotone, subadditive, convex function \( \Psi_\epsilon : \mathbb{R}_+^d \to \mathbb{R} \) such that:

1. **Gradient Stability:** \( \nabla \Psi_\epsilon(x + y) \geq \exp(-\epsilon : \|y\| - \delta) \cdot \nabla \Psi_\epsilon(x) \) coordinate-wise for all \( x, y \in \mathbb{R}_+^d \).

2. **Norm Approximation:** \( \|x\| \leq \Psi_\epsilon(x) \leq \alpha \|x\| + \frac{\gamma}{\epsilon} \) for all \( x \in \mathbb{R}_+^d \).

(We say that \( \Psi_\epsilon \) is a gradient-stable approximation of \( \| \cdot \| \) at scale \( \epsilon \).)

Intuitively, the definition requires the gradient of \( \Psi_\epsilon(x) \) not to decrease drastically in any component when adding a small vector \( y \) to \( x \). For example, the \( \ell_1 \)-norm itself satisfies this property, since it is linear in the non-negative orthant, so its gradient does not change at all. Unfortunately, the vast majority of norms themselves do not fulfill such a property. Consider, for example, the \( \ell_\infty \)-norm, \( \|x\|_\infty := \max_i |x_i| \), which is very “non-linear” in the non-negative orthant: For a small \( \eta_1 \), its gradient at \((1-\eta_1, 1)\) is \((0, 1)\) but its gradient at \((1+\eta_1, 1)\) is \((1, 0)\). However, in this case the standard “smooth” approximation of \( \ell_\infty \) by the softmax function, defined by \( SM_\epsilon(x) := \frac{1}{\epsilon} \ln \left( \sum_i e^{\epsilon x_i} \right) \), is a 0-gradient-stable approximation (see Appendix A.2).

We note that although there are several notions of “smooth” functions in the literature, we are not aware of Definition 1.7 appearing before; see Appendix A.1 for further discussion.

Since gradient-stable functions behave similarly to linear functions, they satisfy many properties that are very useful in the design and analysis of algorithms. In particular, the following two such properties that are central to our \( O(\alpha + \gamma) \)-approximation results.

**Gradient-Stability and a Smooth Game Inequality.** Our analyses of greedy algorithms rely on the fact that a gradient-stable function \( \Psi_\epsilon \) fulfills the following “smooth game inequality”:

\[
\sum_{t=1}^T C(t) \cdot x(t) \leq O\left( (\alpha + \gamma) \cdot \sum_{t=1}^T C(t) \cdot x^* + \alpha \cdot \|1\| \cdot \text{Regret} \right)
\]
Letting $0 = \Lambda^{(0)} \leq \ldots \leq \Lambda^{(T)}$ and $0 = \Lambda^{(0)}_* \leq \ldots \leq \Lambda^{(T)}_*$ be two increasing sequences of vectors, we have
\[
\sum_{t=1}^{T} \left( \Psi(\Lambda^{(t-1)} + y^{(t)}) - \Psi(\Lambda^{(t-1)}) \right) \leq \lambda \Psi(\Lambda^{(T)}) + \mu(\Psi(\Lambda^{(T)}) - \Psi(0)),
\]
where $y^{(t)}_* = \Lambda^{(t)}_* - \Lambda^{(t-1)}_*$ and $\lambda$ and $\mu$ depend on $\epsilon$ and $\delta$ from Definition 1.7. To see why this is useful, consider online generalized load balancing and set $\Lambda^{(t)}$ and $\Lambda^{(t)}_*$ to be the load vectors of our greedy algorithm and the optimum, respectively. By the greedy property, $\Psi(\Lambda^{(t-1)} + y^{(t)}) \leq \Psi(\Lambda^{(t-1)} + y^{(t)}_*)$, and thus
\[
\Psi(\Lambda^{(T)}) - \Psi(0) = \sum_{t=1}^{T} \left( \Psi(\Lambda^{(t-1)} + y^{(t)}) - \Psi(\Lambda^{(t-1)}) \right) \\
\leq \sum_{t=1}^{T} \left( \Psi(\Lambda^{(t-1)} + y^{(t)}_*) - \Psi(\Lambda^{(t-1)}) \right) \leq \lambda \Psi(\Lambda^{(T)}_*) + \mu(\Psi(\Lambda^{(T)}) - \Psi(0)).
\]
This implies
\[
\Psi(\Lambda^{(T)}) \leq \frac{\lambda}{1 - \mu} \Psi(\Lambda^{(T)}_*) + \Psi(0).
\]
Note that if we could follow such an argument when $\Psi$ is the actual norm, then we would immediately get an approximation factor of $\frac{\lambda}{1 - \mu}$, since $\|0\| = 0$. Unfortunately, (1) will usually not hold for norms. As we show, it does hold if $\Psi$ fulfills gradient-stability (see Lemma 3.2). Indeed, for small values of $\epsilon$ and $\delta$, the $\lambda$ and $\mu$ in (1) will get smaller as well. However, this comes at the cost of worse approximation bounds since we also have to control $\Psi(0)$ (i.e., the additive error term $\gamma$ in the approximation).

Let us point out that this general argument has similarities to a typical Price of Anarchy analyses in algorithmic game theory. Particularly, Roughgarden [Rou15] identified a common approach in these analyses and formalized it by the notion of smooth games, which is similar to (1). Many variants of this notion have been discussed, particularly Thang [Tha20] also adapted it to analyze greedy algorithms. However, the main difference is that because we construct the function $\Psi$ ourselves, we can control $\lambda$ and $\mu$, in particular ensuring that $\mu < 1$. This comes at the cost of introducing the $\Psi(0)$ term, which is necessary because it is not possible to construct gradient-stable approximations of (non-linear) norms with $\Psi(0) = 0$.

**Gradient-Stability and a Converse to Jensen’s Inequality.** Another property that gradient-stability brings is that it allows us to approximate $\Psi$ in terms of its gradients. Note that by convexity, namely Jensen’s inequality, we have $\Psi(\Lambda^{(t-1)} + y^{(t)}) - \Psi(\Lambda^{(t-1)}) \geq \langle \nabla \Psi(\Lambda^{(t-1)}), y^{(t)} \rangle$. This way, if we have an increasing sequence $0 = \Lambda^{(0)} \leq \ldots \leq \Lambda^{(T)}$ of vectors, we can lower-bound the increase of $\Psi$ on this sequence by $\Psi(\Lambda^{(T)}) - \Psi(0) \geq \sum_{t=1}^{T} \langle \nabla \Psi(\Lambda^{(t-1)}), y^{(t)} \rangle$, where $y^{(t)} = \Lambda^{(t)} - \Lambda^{(t-1)}$. Gradient-stability implies an approximate converse of this inequality, namely that $\Psi(\Lambda^{(T)}) - \Psi(0)$ can also be approximately upper-bounded in terms of $\sum_{t=1}^{T} \langle \nabla \Psi(\Lambda^{(t-1)}), y^{(t)} \rangle$ if all $y^{(t)}$ are bounded (see Lemma 5.1). In our bandit applications, this lets us linearize the problem and use sublinear-regret properties of classic linear bandit algorithms. Again, the bounds are better if $\epsilon$ and $\delta$ are small, which comes at the cost of a worse approximation of underlying the norm.

The biggest challenge in making this notion of gradient-stability truly useful is designing gradient-stable approximations for norms, which we discuss next.
### 1.5 Constructing Gradient-Stable Approximations of Norms

As pointed out before, most norms themselves do not fulfill gradient-stability. In particular, for every $\ell_p$-norm (except for $p = 1$) the partial derivative of the second coordinate decreases from 1 to almost 0 even for the small shift from $(0, \eta^2)$ to $(\eta, \eta^2)$ when $\eta$ is tiny.

Approximating functions so that gradients change less drastically is indeed an important goal in many fields, e.g., in Online Learning. A standard approach is to add a random perturbation (which is the principle of Follow the Perturbed Leader [KV05]), i.e., define the approximation $\Psi_\epsilon(x) = E_\nu \|x + \nu\|$ for some random noise vector $\nu$. For example, when each coordinate of $\nu$ is an independent Exponential random variable with mean $1/\epsilon$, standard arguments give that regardless of the starting norm $\|\cdot\|$, we have $\nabla \Psi_\epsilon(u + v) \geq e^{-\epsilon\|v\|_\infty} \cdot \nabla \Psi_\epsilon(u)$ for all $u, v \in \mathbb{R}^d$. However, such an approximation will be too weak for our applications. In this case, we get $\Psi_\epsilon(0) = E_\nu \|\nu\| \geq \frac{1}{2}\|1\|$, and so $\gamma \geq \|1\|$ in Definition 1.7 can be as big as $d$, even in case of the $\ell_1$-norm. This makes an $O(\alpha + \gamma)$-approximation in our results very weak, namely it will only be polynomial rather than poly-logarithmic in $d$.

Another natural attempt is to make the noise $\nu$ more adapted to the norm by having it distributed proportional to $e^{-\|x\|}$. In this case, however, we again get $\Psi_\epsilon(0) = d$, which gives an $\Omega(d)$-approximation. (For $\ell_p$-norms actually a “small” deterministic noise suffices, but arguing that requires the specific structure of these norms; see Appendix A.2.) The other standard technique of adding regularization to the variational definition (i.e., $\|x\| = \max_{\|y\|_\star \leq 1} \langle y, x \rangle$ where $\|\cdot\|_\star$ is the dual norm) also suffers from the same issue. We discuss other related notions of gradient-stability in Appendix A.1.

Therefore, one of our main technical contribution is to obtain the right notion of gradient-stability for all monotone symmetric norms. For that, we use Top-k norms as our building blocks, since it is a classic fact (at least since Ky Fan’s Dominance Theorem [HJ12]) that they form a sort of “basis” for all monotone symmetric norms. Since Top-k includes the $\ell_1$ norm, the random perturbation idea alone will not give gradient-stability as discussed above. In order to achieve gradient-stability for Top-k norms, our main idea is to not only use a random perturbation but also to **randomly perturb the parameter** $k$, namely setting $\Psi_\epsilon(x) = E_{K,\nu} \|x + \nu\|_{\text{Top-K}}$. We show that using an appropriately random $K$ with $E[K] \approx k$, it suffices to use $\nu$ having coordinates Exponentially distributed with mean $\frac{1}{k\epsilon}$ (so a factor $k$ smaller than the traditional one). Then it is easy to show $\Psi_\epsilon(x) = E_{K,\nu} \|x + \nu\|_{\text{Top-K}} \leq O(1) \cdot \|x\|_{\text{Top-k}} + \frac{O(\log d)}{\epsilon}$, so that $\alpha = O(1)$ and $\gamma = O(\log d)$. The general idea why this random $K$ helps is as follows: A partial derivative of the Top-K-norm is 1 if and only if this coordinate is among the highest $K$ in the vector, otherwise it is 0; so, if $K$ is deterministic then it is possible to increase only one coordinate of the vector to drastically change the partial derivative; however, randomizing $K$ makes this impossible.

The final step in obtaining a gradient-stable approximation of every symmetric norm is to prove that gradient stability is approximately preserved under compositions. This is useful since we show that any symmetric norm can be approximated (up to an $O(\log d)$ factor) by a composition of Top-k norms. This composition lemma is also of independent interest since it allows us to obtain good approximations for several other online problems where the objective is not symmetric.
2 Gradient-Stability for Symmetric Norms and Compositions

The main result of this section shows that every monotone symmetric norm admits a gradient-stable approximation.

**Theorem 2.1.** Consider a monotone symmetric norm \( \| \cdot \| \) in \( \mathbb{R}^d \). Then for every \( \delta \in (0, 1) \), there is a \( \delta \)-gradient-stable approximation of \( \| \cdot \| \) with error

\[
\left( 2 \cdot e^\delta \cdot (1 + \delta^{-1}) \cdot \log d, \ 4(1 + e^\delta \cdot (1 + \delta^{-1})) \frac{e^\delta \log^2 d}{\delta} \right).
\]

Moreover, given Ball-Optimization oracle access to the norm \( \| \cdot \| \), we can efficiently perform value and gradient oracle calls to \( \delta \)-gradient-stable approximation with a slightly larger error of

\[
\left( 4 \cdot e^\delta \cdot (1 + 4\delta^{-1}) \cdot \log d, \ 16(1 + e^{2\delta} \cdot (1 + \delta^{-1})) \frac{e^\delta \log^2 d}{\delta} \right).
\]

For our applications, it suffices to think of \( \delta \) as a small constant like \( \delta = \frac{1}{4} \), in which case the error becomes \( O(\log d), O(\log^2 d) \).

En route, we give a generic result (Theorem 2.7) on how composition of norms with gradient-stable approximations also admits a gradient-stable approximation; not only this result allows us to obtain gradient-stable approximations for many different norms, but may also be useful in future work.

At a high-level, our approach for proving Theorem 2.1 is to: (a) prove in Section 2.1 that the Top-k norm admits a gradient-stable approximation; (b) prove in Section 2.2 the aforementioned composition result; (c) show in Section 2.3 how to use composition of Top-k norms to obtain a gradient-stable approximation for any symmetric norm.

2.1 Top-k Norms

Recall that for a non-negative vector \( x \in \mathbb{R}^d_+ \), its Top-k norm \( \| x \|_{\text{Top-k}} \) is the sum of the \( k \) largest coordinates of \( x \). We prove the following:

**Theorem 2.2.** For every \( \delta \in (0, 1) \), the Top-k norm in \( \mathbb{R}^d_+ \) admits a \( \delta \)-gradient-stable approximation with error

\[
\left( \exp(\delta) \left( 1 + \frac{1}{\delta} \right), \ \exp(\delta) \frac{H_d}{\delta} \right),
\]

where \( H_d = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{d} \) is the \( d \)-th harmonic number.

We will prove this theorem in the remainder of the section and ignore computational questions. In Appendix A.3 we extend the result to polynomial-time computation using sampling techniques: the resulting approximation is only differentiable almost-everwhere, but this is enough for applications since one can simply add tiny random perturbations to the points where the function is evaluated.

**Theorem 2.3.** For every \( \delta, \rho \in (0, 1) \), there is an algorithm that with probability at least \( 1 - \rho \) constructs an approximation of the Top-k norm in \( \mathbb{R}^d_+ \) with error

\[
\left( \exp(\delta) \left( 1 + \frac{4}{\delta} \right), \ \exp(2\delta) \frac{4H_d}{\delta} \right).
\]
that is differentiable almost-everywhere and wherever it is differentiable it satisfies \( \delta \)-gradient-stability. Moreover, one can perform value and gradient oracle calls in time polynomial in \( \frac{d}{\rho} \).

**Construction of a gradient-stable approximation.** We first define the function \( f_\epsilon : \mathbb{R}^d \rightarrow \mathbb{R} \), which will be our gradient-stable approximation of the Top-\( k \) norm up to a scaling. The function \( f_\epsilon : \mathbb{R}^d \rightarrow \mathbb{R} \) is given by

\[
f_\epsilon(x) := \mathbb{E}_{K,\nu} \|x + \nu\|_{\text{Top-}K}, \quad \text{where}
\]

1. \( \nu_i \) is an independent Exponential random variable with rate \( \lambda = \epsilon k \) for every \( i \in [d] \).
2. \( K \) follows the geometric distribution starting at 1 with parameter \( p = 1 - q \) for \( q = 1 - \frac{1}{k} \), i.e., \( \Pr(K = i) = q^{i-1}(1 - q) \) for \( i \geq 1 \) and \( \mathbb{E}[K] = \frac{k}{q} \).
3. For \( i > d \), we define \( \|x\|_{\text{Top-}i} = \|x\|_{\text{Top-}d} \).

Note that there are two perturbations in \( \|x\|_{\text{Top-}k} \) to obtain \( f_\epsilon(x) \). Firstly, we add a random noise vector \( \nu \). This noise is drawn from an exponential distribution with parameter \( \lambda \) because we have \( \Pr(\nu_i \geq a + b) = e^{-\lambda b} \cdot \Pr(\nu_i \geq a) \) for any non-negative numbers \( a \) and \( b \). So if \( \lambda \) is sufficiently small, the noise \( \nu \) helps “hiding” any increase in \( x \), i.e., it makes \( \nabla f_\epsilon(x + y) \approx \nabla f_\epsilon(x) \). Such a technique is well-established, e.g., it is used in Follow the Perturbed Leader \([KV05]\) or differential privacy \([DR14]\). However, in order to have the desired effect, \( \lambda \) would have to be very small, making the approximation very weak, as discussed in Section 1.5.

Thus, we also perturb by considering the Top-\( K \)-norm for a random \( K \), as opposed to the Top-\( k \)-norm. The \( i \)-th partial derivative of \( f_\epsilon \) now becomes the probability that the \( i \)-th component is among the \( K \) largest entries in \( x + \nu \) (see (3)). If \( K = k \) deterministically, it is possible to slightly increase one entry in \( x \) and change this probability by a lot. With a random \( K \), this is no longer possible. In order to significantly change the probability, many components in \( x \) need to increase.

Indeed, in Lemma 2.5 we will prove that gradients become stable by combining these two perturbations with the respective probability distributions. The idea is as follows: consider any vector \( y \) with \( \|y\|_{\text{Top-}k} \leq 1 \) (for simplicity) by which \( x \) could be shifted. Then there are most \( k \) entries larger than \( \frac{1}{k} \). The random choice of \( K \) ensures that these large entries do not influence the gradient by too much whereas the random choice of \( \nu \) ensures that the effect of the small entries is bounded.

**Guarantees of the function \( f_\epsilon \).** We start by showing that even with the introduction of the two perturbations, \( f_\epsilon \) still gives a good approximation to the Top-\( k \) norm.

In the rest of the section, we denote the function \( f_\epsilon(\cdot) \) by \( f(\cdot) \) for ease of notation.

**Lemma 2.4.** The function \( f \) defined in Equation (2) satisfies

\[
f(x) \geq \exp(-\delta) \cdot \|x\|_{\text{Top-}k} \quad \text{and} \quad f(x) \leq \left( 1 + \frac{1}{\delta} \right) \cdot \|x\|_{\text{Top-}k} + \frac{H_d}{\delta \epsilon}.
\]

Moreover, \( f \) is monotone, subadditive, and convex.
Lemma 2.5. For all \( i \in \{1, \ldots, d\} \), we have
\[
| \nabla_i f(x + y) | \geq \exp(-\epsilon \| y \|_{\text{Top-K}} - \delta) \cdot | \nabla_i f(x) |
\]

Proof. Due to symmetry, it suffices to prove this claim for the first coordinate without loss of generality, i.e., \( i = 1 \). Given a vector \( x \), we use \( x_{(j)} \) to denote its \( j \)-th largest coordinate for \( j \in \{1, \ldots, d\} \) and to denote \( x_{(j)} = 0 \) for \( j > d \). We also use \( x_{-1} \in \mathbb{R}^{d-1} \) to denote the vector \( x \) without its first coordinate.

Notice that the Top-\( j \) norm is differentiable at any point \( x \) with pairwise distinct coordinates and that in this case we have \( \nabla_1 \| x \|_{\text{Top-}j} = 1 \) if the first coordinate is among the highest \( j \) coordinates of \( x \), or equivalently, \( x_1 > (x_{-1})_{(j)} \), and otherwise \( \nabla_1 \| x \|_{\text{Top-}j} = 0 \). Note that the coordinates of \( x + \nu \) will be pairwise distinct almost surely. So, by linearity of expectation, this implies
\[
\nabla_1 f(x) = \sum_{j=1}^{\infty} \Pr(K = j) \cdot \mathbb{E}_\nu \left[ \nabla_1 \| x + \nu \|_{\text{Top-}j} \right]
= \sum_{j=1}^{\infty} \Pr(K = j) \cdot \Pr \left( x_1 + \nu_1 > ((x + \nu)_{-1})_{(j)} \right),
\]
and analogously,
\[
\nabla_1 f(x + y) = \sum_{j=1}^{\infty} \Pr(K = j) \cdot \Pr \left( x_1 + y_1 + \nu_1 > ((x + y + \nu)_{-1})_{(j)} \right).
\]
Recall that we want to show that \( \nabla_1 f(x + y) \) is not much smaller than \( \nabla_1 f(x) \). Comparing the two sums term by term will not be successful because for a fixed \( j \) increasing only a single entry
in $x$ might drastically decrease the probability. Instead, we compare the $j$-th term in the first sum to the $(j+k-1)$-th term in the second sum. By our choice of $K$ being drawn from a geometric distribution with parameter $q$, we have $\Pr(K=j+k-1) = q^{k-1} \Pr(K=j)$, and thus

\[
\nabla_1 f(x+y) \geq \sum_{j=k}^{\infty} \Pr(K=j) \cdot \Pr\left(x_1 + y_1 + \nu_1 > ((x+y+\nu)_{-1})_{(j)} \right)
\]

\[
= \sum_{j=1}^{\infty} \Pr(K=j+k-1) \cdot \Pr\left(x_1 + y_1 + \nu_1 > ((x+y+\nu)_{-1})_{(j+k-1)} \right)
\]

\[
= q^{k-1} \cdot \sum_{j=1}^{\infty} \Pr(K=j) \cdot \Pr\left(x_1 + y_1 + \nu_1 > ((x+y+\nu)_{-1})_{(j+k-1)} \right) .
\] (4)

To conclude the comparison with the right-hand side of (3), we bound the probability involving $((x+y+\nu)_{-1})_{(j+k-1)}$ in terms of that involving $((x+\nu)_{-1})_{(j)}$ using the following claim.

Claim 2.6. For every vector $u \in \mathbb{R}^d_+$ and every vector $v \in \mathbb{R}^d_+$, we have for all $j \geq 1$ that

\[
(u+v)_{(j+k-1)} \leq u_{(j)} + v_{(k)} + \frac{\|v\|_{\text{Top-k}}}{k} .
\]

Proof. We first observe that there are at most than $j - 1$ coordinates $i \in [d]$ for which $u_i > u_{(j)}$. Next, we observe that there are at most $k - 1$ coordinates $i \in [d]$ such that $(u+v)_i > u_{(j)} + v_{(k)}$ but $u_i \leq u_{(j)}$. To see this, suppose otherwise. Then there is a set $S \subseteq [d]$ of size $k$ such that $(u+v)_i > u_{(j)} + v_{(k)}$ and $u_i \leq u_{(j)}$ for all $i \in S$. This implies that $v_i > v_{(k)}$ for every $i \in S$, which is a contradiction.

Combining these two observations, there are no more than $k + j - 2$ coordinates $i \in [d]$ for which $(u+v)_i > u_{(j)} + v_{(k)}$, meaning that $(u+v)_{(j+k-1)} \leq u_{(j)} + v_{(k)}$. $\square$

Now Claim 2.6 implies that

\[
\Pr\left(x_1 + y_1 + \nu_1 > ((x+y+\nu)_{-1})_{(j+k-1)} \right) \geq \Pr\left(x_1 + y_1 + \nu_1 > ((x+\nu)_{-1})_{(j)} + \frac{\|y\|_{\text{Top-k}}}{k} \right) \geq \Pr\left(\nu_1 > ((x+\nu)_{-1})_{(j)} + \frac{\|y\|_{\text{Top-k}}}{k} - x_1 \right) .
\]

Since $\nu_1$ is drawn independently from an exponential distribution with rate $\lambda$, for every pair of non-negative numbers $a, b$ we have $\Pr(\nu_1 \geq a + b) = e^{-\lambda b} \cdot \Pr(\nu_1 \geq a)$. Therefore, the previous inequality becomes

\[
\Pr\left(x_1 + y_1 + \nu_1 > ((x+y+\nu)_{-1})_{(j+k-1)} \right) \geq \exp\left(-\frac{\lambda \|y\|_{\text{Top-k}}}{k}\right) \Pr\left(\nu_1 > ((x+\nu)_{-1})_{(j)} - x_1 \right)
\]

\[
= \exp\left(-\epsilon \|y\|_{\text{Top-k}}\right) \Pr\left(x_1 + \nu_1 > ((x+\nu)_{-1})_{(j)} \right),
\]

where the last equation uses the fact that we set $\lambda = \epsilon k$.

With this at hand, we can employ it on (4) to obtain the gradient lower bound

\[
\nabla_1 f(x+y) \geq q^{k-1} \cdot \exp\left(-\epsilon \|y\|_{\text{Top-k}}\right) \cdot \sum_{j=1}^{\infty} \Pr(K=j) \cdot \Pr\left(x_1 + \nu_1 > ((x+\nu)_{-1})_{(j)} \right)
\]

\[
= q^{k-1} \cdot \exp\left(-\epsilon \|y\|_{\text{Top-k}}\right) \cdot \nabla_1 f(x).
\]

But by our choice of $q$, we have $q^{k-1} = (1 - \delta)k^{k-1} \geq \exp(-\delta)$, which concludes the proof of Lemma 2.5. $\square$
Proof of Theorem 2.2. We can now construct a gradient-stable approximation of Top-k. We claim that the function \( \Psi_\epsilon(\cdot) = \exp(\delta) f_\epsilon(\cdot) \) is a \( \delta \)-gradient-stable approximation of the Top-k norm at scale \( \epsilon \) with the desired error \( (\exp(\delta) (1 + \frac{1}{\delta}), \exp(\delta) \frac{H_d}{\delta} ) \). To see this, first from Lemma 2.4 we see that \( \Psi_\epsilon \) is monotone, subadditive, and convex. Also from this lemma we get the error bound

\[
\| \cdot \|_{\text{Top-k}} \leq \Psi_\epsilon(\cdot) \leq \exp(\delta) \left(1 + \frac{1}{\delta}\right) \| \cdot \|_{\text{Top-k}} + \frac{\exp(\delta) H_d}{\delta \epsilon}.
\]

Finally, from Lemma 2.5 and linearity of the gradient operator, we directly get

\[
\nabla_i \Psi_\epsilon(x + y) \geq \exp(-\epsilon\|y\|_{\text{Top-k}} - \delta) \cdot \nabla_i \Psi_\epsilon(x), \quad \forall x, y \in \mathbb{R}_+^d,
\]

and so \( \Psi_\epsilon \) has all the claimed properties of gradient stability.

Since we can obtain such an approximation for every scale \( \epsilon > 0 \), this proves that the Top-k norm admits the desired gradient-stable approximation. This concludes the proof of the theorem.

2.2 Composition of Norms

We now show that gradient-stability is approximately preserved when considering a “nested” composition of norms. We remark that we assume the normalization that the outer norm \( \| \cdot \|_{(0)} \) is lower bounded by \( \| \cdot \|_{\infty} \) only to obtain a cleaner statement, since this can always be satisfied by an appropriate scaling of the norm.

Theorem 2.7 (Norm Composition). Consider monotone (not necessarily symmetric) “inner” norms \( \| \cdot \|_{(1)}, \ldots, \| \cdot \|_{(\ell)} \) over \( \mathbb{R}^d \), and non-negative matrices \( A_i \in \mathbb{R}^{d \times n} \). Also consider a monotone “outer” norm \( \| \cdot \|_{(0)} \) over \( \mathbb{R}^f \), and assume that \( \| \cdot \|_{\infty} \leq \| \cdot \|_{(0)} \). Then consider the composed norm over \( \mathbb{R}^n \):

\[
norm(x) := \left\| \left( \| A_1 x \|_{(1)}, \ldots, \| A_\ell x \|_{(\ell)} \right) \right\|_{(0)}.
\]

If each norm \( \| \cdot \|_{(i)} \) admits a \( \delta_i \)-gradient-stable approximation with error \( (\alpha_i, \gamma_i) \), then for every \( c > 0 \), \( \text{norm}(\cdot) \) admits a \( \delta \)-gradient-stable approximation with error \( (\alpha, \gamma) \), where:

\[
\alpha = \alpha_0 \left( \max_{i \geq 1} \alpha_i \right), \quad \delta = \delta_0 + \max_{i \geq 1} \delta_i, \quad \gamma = \left( \max_{i \geq 1} \alpha_i + c \right) \cdot \left[ \frac{\alpha_0}{c} (\gamma_1, \ldots, \gamma_\ell) \right]_{(0)} + \gamma_0.
\]

Moreover, given value and gradient oracles to the \( \delta_i \)-gradient-stable approximations of inner and outer norms, we can efficiently simulate value and gradient oracles to this \( \delta \)-gradient-stable approximation.

Its proof is presented in Appendix A.4. In a typical use the \( \alpha_i \)'s and \( \delta_i \)'s are constants, \( c = 1 \), and the outer norm equals \( \| \cdot \|_{\infty} \), in which case we get \( \alpha = \delta = O(1) \) and \( \gamma = O(\max_{i \geq 0} \gamma_i) \); however, the extra flexibility of the parameters will be needed in some of our applications.

As a corollary of the previous lemma, we observe that gradient-stability is invariant with respect to scaling the norm, a fact that will be useful later. This is obtained by applying Theorem 2.7 with a single norm \( \| \cdot \|_{(1)} = \| \cdot \| \) (\( \ell = 1 \)), with \( A_1 \) being the diagonal matrix with \( \beta_i \)'s in the diagonal, \( \| \cdot \|_{(0)} = \| \cdot \|_1 \) (recall that the latter admits a 0-gradient-stable approximation with error \( (1, 0) \), see Lemma A.2), and \( c \rightarrow \infty \).
Lemma 2.8 (Norm Scaling). If the norm $\| \cdot \|$ admit a $\delta$-gradient-stable approximation with error $(\alpha, \gamma)$, then so does the scaled norm $\beta \| \cdot \|$ for every $\beta > 0$ (with the same parameters).

2.3 Symmetric Norms

Given the smoothing for Top-$k$ norms and the composition property above, we now show that every symmetric norm $\| \cdot \|$ can be approximated by a norm $\| \cdot \|$ that is the composition of Top-$k$ norms. Furthermore, this approximation can be computed in polynomial time given Ball-Optimization oracle access to $\| \cdot \|$.

The starting point is that any symmetric monotone norm can be expressed as the supremum of positive combinations of Top-$k$ norms. The following follows from Theorem 1.3 of [LM98] or Lemma 5.2 of [CS19a].

Lemma 2.9. For any symmetric monotone norm $\| \cdot \|$ in $\mathbb{R}^d$, there is a set $W$ of non-negative weights $w \in \mathbb{R}^d_+$ such that for all $x$, we have

$$\|x\| = \max_{w \in W} \sum_{i=1}^d w_i \|x\|_{\text{Top-}i}. \quad (5)$$

While this is not exactly in the format of the composition result from Theorem 2.7, it can be used to obtain an approximation of $\| \cdot \|$ that does have this property.

Lemma 2.10. For any symmetric monotone norm $\| \cdot \|$ in $\mathbb{R}^d$, there are $\log d$ non-negative scalars $c_1, c_2, \ldots, c_{\log d}$ such that the norm

$$\|\|x\|| := \left\| \left( c_1 \|x\|_{\text{Top-}2^1}, \ldots, c_{\log d} \|x\|_{\text{Top-}2^{\log d}} \right) \right\|_{\infty} \quad (6)$$

satisfies $\|x\| \leq \|\|x\|| \leq 2 \log d \cdot \|x\|$.

Note that a similar property has been proved in [CS19a]. See Remark 2.12 for a discussion.

Proof. Let $W$ be the set given by the previous lemma applied to the norm $\| \cdot \|$. First, using a bucketing strategy, we sparsify the sums appearing in (5) so that each has $\log d$ terms, losing a factor of 2.

More precisely, consider a $w \in W$, and the sum $\sum_{i=1}^d w_i \|x\|_{\text{Top-}i}$. Round up each $i$ to the nearest power of 2, let $B_j$ be all the $i$’s that get rounded to the power $2^j$. Notice that when we round $i$ to the nearest power, say $i'$, then the Top-$i$ vs Top-$i'$ changes by at most a factor of 2. This gives that

$$\sum_{i=1}^d w_i \|x\|_{\text{Top-}i} \leq \sum_{j=1}^{\log d} \left( \sum_{i \in B_j} w_i \|x\|_{\text{Top-}2^j} \right) \quad \text{and} \quad \sum_{i=1}^d w_i \|x\|_{\text{Top-}i} \geq \frac{1}{2} \sum_{j=1}^{\log d} \left( \sum_{i \in B_j} w_i \|x\|_{\text{Top-}2^j} \right).$$

Then define $w'$ as $w'_j := \sum_{i \in B_j} w_i$, and add it to the set $W'$. 

14
Repeating this operation for all \( w \in W \) gives a set \( W' \subseteq \mathbb{R}^{\log d} \) that provides the desired “sparse” approximation to \( \| \cdot \| \) (a similar inequality has been used in [CS19a]):

\[
\| x \| \leq \max_{w' \in W'} \sum_{j \leq \log d} w'_j \| x \|_{\text{Top-} 2^j} \leq 2 \| x \| .
\]

Moreover, since each sum in the previous expression has only \( \log d \) terms, replacing each sum by a max only loses a \( \log d \) factor, namely:

\[
\frac{1}{\log d} \cdot \| x \| \leq \max_{w' \in W'} \max_{j \leq \log d} w'_j \| x \|_{\text{Top-} 2^j} \leq 2 \| x \| .
\]

Setting \( c_j := \log d \cdot \max_{w' \in W'} w'_j \) gives the desired result.

With all these elements, we can finally prove that every monotone symmetric norm admits a gradient-stable approximation.

Proof of Theorem 2.1. From (6) we see that \( \| \| \| \) is a composition of norms covered by Theorem 2.7 with the matrices \( A_i \) being the identity; thus, we will employ it to obtain the desired gradient-stable approximation of \( \| \| \) (and then of the original norm \( \| \cdot \| \)).

More precisely, fix \( \delta \in (0, 1) \) and let \( \tilde{\alpha} = e^\delta (1 + \frac{1}{\delta}) \) and \( \tilde{\gamma} = \exp(\delta) \frac{H_d}{\delta} \) be the parameters from Theorem 2.2. Combining the gradient-stable approximation of Top-\( k \) norms (Theorem 2.2) and the invariance with respect to scaling the norms (Lemma 2.8), we get that the scaled norm \( c_j \| \cdot \|_{\text{Top-} 2^j} \) that appears in \( \| \| \) admits a \( \delta \)-gradient-stable approximation with error \( (\tilde{\alpha}, \tilde{\gamma}) \). Also recall that the norm \( \| \cdot \|_\infty \) with \( (\log d) \)-coordinates admits a 0-gradient-stable approximation with error \( (1, \ln \log d) \) (Lemma A.1). Thus, applying composition Theorem 2.7 with (parameter \( c = 1 \)) we get that \( \| \| \) admits a \( \delta \)-gradient-stable approximation with error \( (\alpha, \gamma) \), where \( \alpha = \tilde{\alpha} \) and \( \gamma = (\tilde{\alpha} + 1) \cdot \left[ \tilde{\gamma} + \ln \log d \right] \leq 2(\tilde{\alpha} + 1) \cdot \frac{e^\delta \log d}{\delta} .
\]

We claim that this gives the desired gradient-stable approximation of the original norm \( \| \cdot \| \). For that, let \( h \) be a \( \delta \)-gradient-stable approximation of \( \| \| \) with error \( (\alpha, \gamma) \) at scale \( \epsilon \). Then this is a \( \delta \)-gradient-stable approximation of \( \| \cdot \| \) with error \( (2\alpha \log d, 2\gamma \log d) \) at scale \( \epsilon' = 2\epsilon \log d \): the error is

\[
\| \cdot \| \leq \| \| \| \| \leq h(\cdot) \leq \alpha \| \| + \| \| \frac{\gamma}{\epsilon} \leq 2\alpha (\log d) \| \cdot \| + \frac{2\gamma \log d}{\epsilon'},
\]

and the gradient stability

\[
\nabla h(u + v) \geq e^{-\epsilon \| v \| - \delta} \cdot \nabla h(v) \geq e^{-\epsilon' \| v \| - \delta} \cdot \nabla h(u), \quad \forall u, v \in \mathbb{R}^d .
\]

Since this construction holds for every \( \epsilon' > 0 \), the norm \( \| \cdot \| \) admits the desired approximation. Unpacking the values of \( \alpha \) and \( \gamma \) concludes the existential part of the proof of Theorem 2.1.
Computation. Note that if for a symmetric norm $\| \cdot \|$ we are explicitly given $W$ in (5), then in time polynomial in $d$ and $|W|$ we can obtain the $\delta$-gradient stable approximation $h$ of $\| \cdot \|$ with error $(\alpha, \gamma)$ at scale $\epsilon$ using the computational part of Theorem 2.2 and Theorem 2.7. If instead, we are only given Ball-Optimization oracle access to $\| \cdot \|$, we can reduce the problem to one with a poly-size $W$ using the following lemma.

Lemma 2.11. For any symmetric monotone norm $\| \cdot \|$ in $\mathbb{R}^d$ and every $\epsilon' > 0$, there is a set $W'$ of non-negative weights $w \in \mathbb{R}^d_+$ with $|W'| \leq \left( \frac{d}{\epsilon'} \right)^{O(1/\epsilon')}$ that can be computed using Ball-Optimization oracle access to $\| \cdot \|$ in time $\left( \frac{d}{\epsilon'} \right)^{O(1/\epsilon')}$ such that for all $x \in \mathbb{R}^d_+$ we have

$$
(1 - \epsilon') \| x \| \leq \max_{w \in W'} \sum_{i=1}^{d} w_i \| x \|_{\text{Top-i}} \leq (1 + \epsilon') \| x \|.
$$

This lemma has essentially been shown in [CS19a] as Theorem 5.4. For completeness, we provide the proof in Appendix A.5 in a simplified version with an improved bound.

Using this lemma, we obtain the computational result of Theorem 2.1 by first computing the set $W'$ and then applying the above approximation using Theorem 2.3 on the norm $f$ defined by $f(x) = \max_{w \in W'} \sum_{i=1}^{d} w_i \| x \|_{\text{Top-i}}$. In combination, we only lose a constant factor compared to the existential result.

Remark 2.12. Another approach to prove Theorem 2.1 is to use a different construction for the log $d$-approx $\| \cdot \|$ of $\| \cdot \|$ in Lemma 2.10 where the outer $\ell_\infty$ norm is replaced with an $\ell_1$ norm. (Such an $\ell_1$ approx has been previously given in [CS19a]s.) Although this approach has the advantage that the proof of the composition theorem for $\ell_1$ is much easier than the general Theorem 2.7, it loses an extra log $d$ factor in the competitive ratio since the $\gamma$ parameter adds up for $\ell_1$. Moreover, our general composition theorem will play a crucial role in some of our applications later where the outer norm is not $\ell_1$.

3 Gradient-Stability implies a Smooth Game Inequality

In this section we prove a smooth game inequality for gradient-stable norm approximations. As discussed in Section 1.4, this will play a central role in our design of both online and bandit algorithms. Before showing the inequality, we prove another important “approximate supermodularity” property satisfied by gradient-stable approximations.

Lemma 3.1 (Approximate Supermodularity). If $\Psi_\epsilon$ is a $\delta$-gradient-stable approximation of a norm $\| \cdot \|$ at scale $\epsilon$, then for any vectors $\Lambda, y, z \in \mathbb{R}^d_+$ it satisfies

$$
\Psi_\epsilon(\Lambda + y + z) - \Psi_\epsilon(\Lambda + z) \geq \exp(-\epsilon \cdot \| z \| - \delta) \cdot \left( \Psi_\epsilon(\Lambda + y) - \Psi_\epsilon(\Lambda) \right).
$$

Proof. Using gradient-stability and non-negativity of $\nabla \Psi_\epsilon$, we have that

$$
\Psi_\epsilon(\Lambda + y + z) - \Psi_\epsilon(\Lambda + z) = \int_{0}^{1} \langle \nabla \Psi_\epsilon(\Lambda + z + \tau y), y \rangle \, d\tau
$$
\begin{equation*}
\geq \exp(-\epsilon \cdot \|z\| - \delta) \cdot \int_{0}^{1} \langle \nabla \Psi_{\epsilon}(\Lambda + \tau y) , y \rangle \, d\tau = \exp(-\epsilon \cdot \|z\| - \delta) \cdot (\Psi_{\epsilon}(\Lambda + y) - \Psi_{\epsilon}(\Lambda)).
\end{equation*}

We use approximate supermodularity to prove the following smoothness.

**Lemma 3.2** (Smooth Game Inequality). Let \(0 = \Lambda^{(0)} \leq \Lambda^{(1)} \leq \ldots \leq \Lambda^{(T)}\) and \(0 = \Lambda^{(0)}_{\ast} \leq \Lambda^{(1)}_{\ast} \leq \ldots \leq \Lambda^{(T)}_{\ast}\) be any two non-decreasing sequences of non-negative vectors and let \(y^{(t)} = \Lambda^{(t)} - \Lambda^{(t-1)}\) and \(y^{(t)}_{\ast} = \Lambda^{(t)}_{\ast} - \Lambda^{(t-1)}_{\ast}\). Then we have

\[
\sum_{t=1}^{T} \left( \Psi_{\epsilon}(\Lambda^{(t-1)} + y^{(t)}_{\ast}) - \Psi_{\epsilon}(\Lambda^{(t-1)}) \right) \leq \zeta \Psi_{\epsilon}(\Lambda^{(T)}_{\ast}) + (\zeta - 1) \left( \Psi_{\epsilon}(\Lambda^{(T)}) - \Psi_{\epsilon}(\Lambda^{(0)}) \right),
\]

where \(\zeta = \exp\left( \epsilon \|\Lambda^{(T)}_{\ast}\| + \delta \right)\).

**Proof.** The idea is to consider the sequence \(\Lambda^{(t)} + \Lambda^{(t)}_{\ast}\) in which we add both \(y^{(t)}\) and \(y^{(t)}_{\ast}\) in every step. The increases of \(\Psi_{\epsilon}\) can then be related using approximate supermodularity and subadditivity. We write

\[
\Psi_{\epsilon}(\Lambda^{(t)} + \Lambda^{(t)}_{\ast}) - \Psi_{\epsilon}(\Lambda^{(t-1)} + \Lambda^{(t)}_{\ast}) = \left( \Psi_{\epsilon}(\Lambda^{(t)} + \Lambda^{(t)}_{\ast}) - \Psi_{\epsilon}(\Lambda^{(t-1)} + \Lambda^{(t)}_{\ast}) \right) + \left( \Psi_{\epsilon}(\Lambda^{(t-1)} + \Lambda^{(t)}_{\ast}) - \Psi_{\epsilon}(\Lambda^{(t-1)} + \Lambda^{(t-1)}_{\ast}) \right).
\]

Furthermore, by approximate supermodularity property in Lemma 3.1, we have

\[
\Psi_{\epsilon}(\Lambda^{(t)} + \Lambda^{(t)}_{\ast}) - \Psi_{\epsilon}(\Lambda^{(t-1)} + \Lambda^{(t)}_{\ast}) \geq \exp\left( -\epsilon \|\Lambda^{(t)}_{\ast}\| - \delta \right) \left( \Psi_{\epsilon}(\Lambda^{(t)}) - \Psi_{\epsilon}(\Lambda^{(t-1)}) \right)
\]

and

\[
\Psi_{\epsilon}(\Lambda^{(t-1)} + \Lambda^{(t)}_{\ast}) - \Psi_{\epsilon}(\Lambda^{(t-1)} + \Lambda^{(t-1)}_{\ast}) \geq \exp\left( -\epsilon \|\Lambda^{(t-1)}_{\ast}\| - \delta \right) \left( \Psi_{\epsilon}(\Lambda^{(t-1)}) + y^{(t)}_{\ast} - \Psi_{\epsilon}(\Lambda^{(t-1)}) \right).
\]

By monotonicity of the norm, we have \(\|\Lambda^{(t-1)}_{\ast}\| \leq \|\Lambda^{(T)}_{\ast}\|\) as well as \(\|\Lambda^{(t)}_{\ast}\| \leq \|\Lambda^{(T)}_{\ast}\|\). So, we get

\[
\Psi_{\epsilon}(\Lambda^{(t)} + \Lambda^{(t)}_{\ast}) - \Psi_{\epsilon}(\Lambda^{(t-1)} + \Lambda^{(t)}_{\ast}) \geq \frac{1}{\zeta} \left( \Psi_{\epsilon}(\Lambda^{(t-1)} + y^{(t)}_{\ast}) - \Psi_{\epsilon}(\Lambda^{(t-1)}) + \Psi_{\epsilon}(\Lambda^{(t)}) - \Psi_{\epsilon}(\Lambda^{(t-1)}) \right).
\]

Taking the sum over all \(t\), two of the sums telescope, so we get

\[
\Psi_{\epsilon}(\Lambda^{(T)} + \Lambda^{(T)}_{\ast}) - \Psi_{\epsilon}(\Lambda^{(0)}) \geq \frac{1}{\zeta} \left( \sum_{t=1}^{T} \left( \Psi_{\epsilon}(\Lambda^{(t-1)} + y^{(t)}_{\ast}) - \Psi_{\epsilon}(\Lambda^{(t-1)}) \right) + \Psi_{\epsilon}(\Lambda^{(T)}) - \Psi_{\epsilon}(\Lambda^{(0)}) \right).
\]

By subadditivity of \(\Psi_{\epsilon}\), we furthermore have \(\Psi_{\epsilon}(\Lambda^{(T)} + \Lambda^{(T)}_{\ast}) \leq \Psi_{\epsilon}(\Lambda^{(T)}) + \Psi_{\epsilon}(\Lambda^{(T)}_{\ast})\). In combination, this proves the lemma. \(\square\)
4 Applications to Online Algorithms

In this section we use the smooth game inequality to analyze our online algorithms.

4.1 Online Generalized Load Balancing

Recall the Online Generalized Load Balancing problem (GenLoadBal) defined in the introduction. There are $m$ machines, and $T$ jobs come one-by-one. In each time step $t \in [T]$ a cost matrix $C(t) \in \mathbb{R}^{m \times k}_+$ comes, indicating the $k$ processing options for this job. After seeing $C(t)$ the goal is to immediately make a scheduling decision $x(t) \in \{e_1, \ldots, e_k\}$ and incur a load vector $C(t)x(t)$. To measure the quality of the solution, it is given a norm $\| \cdot \|$ over $\mathbb{R}^m$, and the goal is to minimize the norm of the total load vector $\Lambda(T) := \sum_{t=1}^{T} C(t)x(t)$. Let $OPT$ denote the cost of the optimal offline solution.

The following gives a general result for this problem.

**Theorem 4.1.** Consider the problem GenLoadBal with a norm $\| \cdot \|$. Fix $\delta = \frac{1}{4}$, and suppose $\| \cdot \|$ has gradient-stable approximations with uniform error $(\alpha, \gamma)$. Then there exists an $O(\alpha + \gamma)$-competitive algorithm for this problem. Moreover, this algorithm is efficient given value oracle access to the gradient-stable approximation.

Together with the gradient-stable approximation of the $\ell_p$ norms from Lemma A.2, this gives an $O(\min\{p, \log m\})$ competitive ratio in this special case. Instead, using the approximation of general monotone symmetric norms from Theorem 2.1 gives an $O(\log^2 m)$ competitive ratio in this general case, which proves Theorem 1.2.

**Proof of Theorem 4.1.** The algorithm that achieves the guarantee is the following: First, by the standard guess-and-double trick we assume without loss of generality that we have an estimate $\hat{OPT}$ within a factor of two of the offline optimum $OPT$, namely $\hat{OPT} \in [OPT, 2OPT]$ [AAF93]. Given this estimate, the algorithm takes a $\delta$-gradient-stable approximation $\Psi = \Psi_\epsilon$ of $\| \cdot \|$ with parameters $\epsilon = \frac{1}{4OPT}$ and $\delta = \frac{1}{4}$ and make scheduling decisions greedily with respect to $\Psi$.

Observe that this algorithm requires only $k$ calls to the value oracle of $\Psi$ per time step.

**Algorithm 1** Gradient-Stable Greedy

```python
input: Estimate $\hat{OPT} \in [OPT, 2OPT]$
Let $\Psi = \Psi_\epsilon$ be a $\frac{1}{4}$-gradient-stable approximation of $\| \cdot \|$ with error $(\alpha, \gamma)$ for $\epsilon = \frac{1}{4OPT}$.
for each time $t$ do
    Choose $x(t) = \arg\min_{x \in \{e_1, \ldots, e_k\}} \Psi(C^{(1)}x(1) + \ldots + C^{(t-1)}x(t-1) + C(t)x)$.
end for
```

We prove that this algorithm is $O(\alpha + \gamma)$-competitive, which implies Theorem 4.1. For ease of notation, we define $y(t) := C(t)x(t)$ to be the cost vector incurred by the algorithm in step $t$, and $\Lambda(t) := y(1) + \ldots + y(t)$ to be the cumulative cost vector. Moreover, let $y_*^t$ be the cost vector of the offline optimal solution at step $t$, and define $\Lambda_*^t := y_*^1 + \ldots + y_*^t$.
We first upper bound the total cost of the algorithm $\Psi(\Lambda^{(T)})$ with respect to the proxy function $\Psi$. Since we run a greedy algorithm, we have
\[
\Psi(\Lambda^{(T)}) - \Psi(0) = \sum_{t=1}^{T} \left( \Psi(\Lambda^{(t)}) - \Psi(\Lambda^{(t-1)}) \right) \leq \sum_{t=1}^{T} \left( \Psi(\Lambda^{(t-1)} + y_{*}^{(t)}) - \Psi(\Lambda^{(t-1)}) \right).
\]

The smooth game inequality in Lemma 3.2 implies that for $\zeta = \exp \left( \epsilon \|\Lambda^{(T)}\|_* + \delta \right)$, we have
\[
\sum_{t=1}^{T} \left( \Psi(\Lambda^{(t-1)} + y_{*}^{(t)}) - \Psi(\Lambda^{(t-1)}) \right) \leq \zeta \Psi(\Lambda^{(T)}) + (\zeta - 1) \left( \Psi(\Lambda^{(T)}) - \Psi(0) \right).
\]

So, in combination,
\[
(2 - \zeta) \left( \Psi(\Lambda^{(T)}) - \Psi(0) \right) \leq \zeta \Psi(\Lambda^{(T)}),
\]
or equivalently
\[
\Psi(\Lambda^{(T)}) \leq \frac{\zeta}{2 - \zeta} \Psi(\Lambda^{(T)}_*) + \Psi(0).
\]

Now using the approximation properties of $\Psi$, we can translate this into an upper bound of the true cost of the algorithm with respect to the original norm $\| \cdot \|$. That is, by definition of $\Psi$ we have $\Psi(\Lambda^{(T)}) \geq \|\Lambda^{(T)}\|$, $\Psi(\Lambda^{(T)}_*) \leq \alpha \|\Lambda^{(T)}_*\| + \frac{\gamma}{\epsilon}$, and $\Psi(\Lambda^{(0)}) \leq \frac{\gamma}{\epsilon}$. Therefore,
\[
\text{Alg’s cost} = \|\Lambda^{(T)}\| \leq \frac{\zeta}{2 - \zeta} \left( \alpha \|\Lambda^{(T)}_*\| + \frac{\gamma}{\epsilon} \right) + \frac{\gamma}{\epsilon}.
\]

Since $\epsilon = \frac{1}{4\text{OPT}}$ and $\delta = \frac{1}{4}$, we have $\zeta = \exp \left( \frac{\text{OPT}}{4\text{OPT}} + \frac{1}{4} \right) \leq \exp \left( \frac{1}{2} \right) < 1.7$. So,
\[
\text{Alg’s cost} = \|\Lambda^{(T)}\| \leq O \left( \alpha \|\Lambda^{(T)}_*\| + \gamma \frac{\text{OPT}}{\epsilon} \right) \leq O \left( (\alpha + \gamma) \cdot \text{OPT} \right).
\]

This proves the approximation ratio of the algorithm, and implies Theorem 4.1.

Next, we see applications of our techniques to online algorithms with non-symmetric objectives.

### 4.2 Online Vector Scheduling

In this section we consider Online Vector Scheduling problem (VecSched), which is related to GenLoadBal but has a non-symmetric objective. In this problem we have $m$ machines with each machine having $r$ resources (coordinates). At time $t$, a job comes and reveals the vector load $\ell_{t,i} \in \mathbb{R}^r_+$ it incurs if assigned to machine $j$. At this point, the algorithm needs to select a machine $\sigma(t) \in [m]$ where this job is assigned. For each resource $k$, there is an “inner” norm $\| \cdot \|_{(k)}$ over $\mathbb{R}^m$ (used to aggregate the cost for this recourse over all machines), and the “outer” norm $\ell_\infty$ over $\mathbb{R}^r$ (used to aggregate the cost over all resources). Letting $\text{load}(k) := (\sum_{t: \sigma(t)=i} \ell_{t,i}) \in \mathbb{R}^m_+$ denote the vector load incurred on resource $k$ by the algorithm, the goal is to minimize
\[
\left\| \left( \|\text{load}(1)\|_{(1)}, \ldots, \|\text{load}(r)\|_{(r)} \right) \right\|_\infty.
\]

We show that the algorithm for GenLoadBal can be used to obtain the following.
**Theorem 4.2.** Consider the problem VecSched with inner norms $\| \cdot \|_{(1)}, \ldots, \| \cdot \|_{(r)}$. If for every $k \in [r]$ the norm $\| \cdot \|_{(k)}$ has a $\frac{1}{2}$-gradient-stable approximation with error $(\alpha_k, \gamma_k)$, then there exists an algorithm for the problem with competitive ratio

$$O \left( \max_k \gamma_k + (\log r) \max_k \alpha_k \right).$$

Moreover, this algorithm is efficient given value and gradient oracle access to these $\frac{1}{2}$-gradient-stable approximations to norms.

For the special case where the inner norms are all $\ell_p$’s this obtains an $O(\log m + \log r)$ competitive ratio, and for the more general case of monotone symmetric norms this gives an $O(\log^2 m + \log r)$-approximation for general monotone symmetric inner norms. In particular this proves Theorem 1.3.

**Proof of Theorem 4.2.** The idea is to embed VecSched into an instance of GenLoadBal with a nested norm objective, and use the algorithm for the latter from Theorem 4.1. More precisely, given an instance $I_{VS}$ of VecSched we construct the following instance $I_{GLB}$ of GenLoadBal: $I_{GLB}$ has $m \cdot r$ machines, each indexed by $(i, k) \in [m] \times [r]$ (which corresponds to the $k$-th resource of machine $i$ in $I_{VS}$), and each job has $m$ processing options (corresponding to the assignment on one of the machines of $I_{VS}$). The processing option $i$ for job $t$ adds load $(\ell_{t,i})_k$ on machine $(i, k)$ for all $k \in [r]$, i.e., $C_{t,(i,k)} = (\ell_{t,i})_k$ for all $k \in [r]$. Finally, this instance $I_{GLB}$ has as objective function the norm $\| \cdot \|$ defined as

$$\| x \| := \left\| \left( \| A_1 x \|_{(1)}, \ldots, \| A_r x \|_{(r)} \right) \right\|_{\infty},$$

where $A_k$ is the matrix that projects a vector $x \in \mathbb{R}^{m \cdot r}$ (indexed by $(i, k) \in [m] \times [r]$) to the $m$-dimensional subvector vector $A_k x = (x_{(i,k)})_i$.

By construction, scheduling job $t$ in machine $i$ in the original instance $I_{VS}$ corresponds precisely to choosing the processing option $i$ for job $t$ in the new instance $I_{GLB}$. Moreover, the corresponding schedules on the instances have the same cost with respect to the objective functions of their respective instances. Thus, the online algorithm for approximating $I_{VS}$ is simply to construct the instance $I_{GLB}$ in an online fashion (which is easy to do), and to apply over it the algorithm from Theorem 4.1 to decide the processing options/assignments of the jobs.

For its competitive ratio, it suffices to show that $\| \cdot \|$ admits a good gradient-stable approximation. Since $\| \cdot \|$ is a composition of norms, we use Theorem 2.7: Recalling that the norm $\| \cdot \|_{\infty}$ over $r$ coordinates admits a 0-gradient-stable approximation with error $(1, \ln r)$ (Lemma A.1) and using the assumption on the norms $\| \cdot \|_{(k)}$’s, we can employ Theorem 2.7 with $c = \max_k \alpha_k$ to directly get that $\| \cdot \|$ admits a $\frac{1}{4}$-gradient-stable approximation with error $(\alpha, \gamma)$, where

$$\alpha = \max_k \alpha_k \quad \text{and} \quad \gamma = 2(\max_k \alpha_k) \left( \frac{1}{\max_k \alpha_k} \| \gamma_1, \ldots, \gamma_r \|_{\infty} + \ln r \right) = 2 \max_k \gamma_k + 2(\ln r) \max_k \alpha_k.$$

Given the guarantee from Theorem 4.1, our algorithm for VecSched has competitive ratio $O(\alpha + \gamma) = O(\max_k \gamma_k + (\ln r) \max_k \alpha_k)$.

Finally, observe that Theorem 2.7 also implies that we have efficient access to the value and gradient oracles of the above $\frac{1}{4}$-gradient-stable approximation of $\| \cdot \|$. Thus, we can run the algorithm in Theorem 4.1 efficiently. This concludes the proof of Theorem 4.2. 


4.3 Online Generalized Assignment with Convex Costs

Next we consider the Online Generalized Assignment with Convex Costs problem (ONGAP) where the objective function is not directly a norm. In this problem there are $m$ machines and $T$ online jobs, each with $k$ processing options. However, each job $t$ has both a cost matrix $C^{(t)}$ and a duration matrix $D^{(t)}$, both $m \times k$, indicating the cost and duration vector incurred over the machines if we process the job with each of the $k$ options. There are two non-negative functions $f_{\text{cost}}, f_{\text{dur}} : \mathbb{R}^m \to \mathbb{R}_+$ that are convex, monotone, and symmetric. The goal is to choose the processing options for the jobs $x^{(1)}, \ldots, x^{(T)} \in \{e_1, \ldots, e_k\}$ to minimize the sum of total cost and duration:

$$f_{\text{cost}} \left( \sum_t C^{(t)} x^{(t)} \right) + f_{\text{dur}} \left( \sum_t D^{(t)} x^{(t)} \right).$$

Also, recall that a function $f$ has growth of order at most $p$ if $f(\alpha x) \leq \alpha^p f(x)$ for all $\alpha \geq 1$ and for all vectors $x \in \mathbb{R}^m_+$.

We show that we can get an $O(\log^{2p} m)$-competitive algorithm for this problem, where $p$ is an upper bound on the growth of $f_{\text{cost}}$ and $f_{\text{dur}}$.

**Theorem 1.4.** Consider the problem ONGAP with functions $f_{\text{cost}}, f_{\text{dur}}$ that are convex, monotone, and symmetric. Then there is an $O(\log^{2p} m)$-competitive algorithm for this problem, where $p$ is the maximum of the growth order of $f_{\text{cost}}$ and $f_{\text{dur}}$. Moreover, this algorithm is efficient given Ball-Optimization access to the norms $\| \cdot \|_{f_{\text{cost}}}$ and $\| \cdot \|_{f_{\text{dur}}}$, where for a convex function $f$ its norm is $\| x \|_f := \inf \{ \lambda > 0 \mid f(\hat{x}) \leq 1 \}$.

In the remainder of the section we describe the algorithm and then prove this result. Again, using the standard guess-and-double trick we can assume that an estimate $\hat{\text{OPT}} \in [\text{OPT}, 2 \text{OPT}]$ of the offline optimum $\text{OPT}$ is given (this only adds an extra constant factor to the competitive ratio). The high-level idea is to convert the objective function into a composition of norms associated with the sub-level sets of the functions $f_{\text{cost}}, f_{\text{dur}}$ of value at most $\hat{\text{OPT}}$, and then show that this composition admits a gradient-stable approximation so that we can apply the GenLoadBal algorithm from Theorem 4.1.

More precisely, given an instance $I_{\text{ongap}}$, we construct the instance $I_{\text{GLB}}$ as follows: $I_{\text{GLB}}$ has $2m$ machines, each indexed by $(i, w) \in [m] \times [2]$, and each job has $k$ processing options. The processing option $j$ for job $t$ adds load column $(C^{(t)}, j) \in \mathbb{R}_{\geq 0}^m$ to the machines $((i, 1))_i$ and adds load column $(D^{(t)}, j) \in \mathbb{R}_{\geq 0}^m$ to the machines $((i, 2))_i$. Define the norms

$$\| x \|_{\text{cost}} := \inf \{ \lambda > 0 \mid f_{\text{cost}} \left( \frac{X}{\lambda} \right) \leq \hat{\text{OPT}} \} \quad \text{and} \quad \| x \|_{\text{dur}} := \inf \{ \lambda > 0 \mid f_{\text{dur}} \left( \frac{X}{\lambda} \right) \leq \hat{\text{OPT}} \}.$$

Finally, the instance $I_{\text{GLB}}$ has objective function the norm $\| \cdot \|$ defined as

$$\| x \| = \left\| \left( \| A_1 x \|_{\text{cost}} , \| A_2 x \|_{\text{dur}} \right) \right\|_1,$$

where as before $A_w$ is the matrix that projects a vector $x \in \mathbb{R}^{2m}$ (indexed by $(i, w) \in [m] \times [2]$) to the $m$-dimensional subvector vector $A_w x = (x_{i(w)})_i$. 

21
The online algorithm for approximating $\mathcal{I}_{\text{onGAP}}$ is simply to construct the instance $\mathcal{I}_{\text{GLB}}$ above in an online fashion (which is easy to do), and to apply over it the algorithm from Theorem 4.1 to decide the processing options of the jobs.

We prove that this algorithm is $O(\log^2 p m)$-competitive, as claimed in Theorem 1.4.

**Proof of Theorem 1.4.** Let $x^{(1)}, \ldots, x^{(T)}$ and $x_s^{(1)}, \ldots, x_s^{(T)}$ be the solution returned by the algorithm and the optimal offline solution for the instance $\mathcal{I}_{\text{onGAP}}$ (with value $\text{OPT}$), respectively. Let $\Lambda(T) = \sum_t O(t) x^{(t)}$ and $M(T) = \sum_t D(t) x^{(t)}$, and define $\Lambda_s(T)$ and $M_s(T)$ similarly with respect to the optimal solution.

Since $f_{\text{cost}}(\Lambda_s(T)) + f_{\text{dur}}(M_s(T)) = \text{OPT} \leq \widehat{\text{OPT}}$ and the functions are non-negative, we have that $\|\Lambda_s(T)\|_{\text{cost}}$ and $\|M_s(T)\|_{\text{dur}}$ are at most 1. Therefore, we have

$$\left\| \left( \|\Lambda_s(T)\|_{\text{cost}}, \|M_s(T)\|_{\text{dur}} \right) \right\|_1 \leq 2,$$

and since the left-hand side is the value of the solution $x_s$ in the instance $\mathcal{I}_{\text{GLB}}$, the optimum of the latter at most 2.

Moreover, since the functions $f_{\text{cost}}$ and $f_{\text{dur}}$ are symmetric, the norms $\|\cdot\|_{\text{cost}}$ and $\|\cdot\|_{\text{dur}}$ are also symmetric. Therefore, from Theorem 2.1 we know that both admit a $\frac{1}{4}$-gradient-stable approximation with error $(O(\log m), O(\log^2 m))$. Also, from Lemma A.2 the $\ell_1$-norm admits a 0-gradient-stable approximation with error $(1, 0)$. Therefore employing Theorem 2.7 (with parameter $c = O(\log m)$) to the norm $\|\cdot\|$ we see that it admits a $\frac{1}{4}$-gradient-stable approximation with error $(\alpha, \gamma)$ where

$$\alpha = O(\log m) \quad \text{and} \quad \gamma = O(\log m) \left[ \frac{1}{O(\log m)} \cdot 2 \cdot O(\log^2 m) \right] = O(\log^2 m).$$

Moreover, since Theorem 2.1 gives us efficient value and gradient oracles for $\frac{1}{4}$-gradient-stable approximations of both $\|\cdot\|_{\text{cost}}$ and $\|\cdot\|_{\text{dur}}$, we also get from Theorem 2.7 efficient value and gradient oracles for $\frac{1}{4}$-gradient-stable approximation of $\|\cdot\|$.

Therefore, by the guarantee from Theorem 4.1, the solution $x^{(1)}, \ldots, x^{(T)}$ is $O(\alpha + \gamma) = O(\log^2 m)$ competitive for the instance $\mathcal{I}_{\text{GLB}}$, and so it has $\|\cdot\|$-value at most $2 \cdot O(\log^2 m) = O(\log^2 m)$, namely

$$\|\Lambda(T)\|_{\text{cost}} + \|M(T)\|_{\text{dur}} = \left\| \left( \|\Lambda(T)\|_{\text{cost}}, \|M(T)\|_{\text{dur}} \right) \right\|_1 \leq O(\log^2 m).$$

Unpacking the definitions, this implies in particular that

$$f_{\text{cost}} \left( \frac{\Lambda(T)}{O(\log^2 m)} \right) \leq \widehat{\text{OPT}} \leq 2\text{OPT}.$$

Then using the fact that the function $f_{\text{cost}}$ has growth of order at most $p$, this implies

$$f_{\text{cost}}(\Lambda(T)) = f_{\text{cost}} \left( O(\log^2 m) \cdot \frac{\Lambda(T)}{O(\log^2 m)} \right) \leq O(\log^2 m)^p \cdot f_{\text{cost}} \left( \frac{\Lambda(T)}{O(\log^2 m)} \right) \leq O(\log^{2p} m) \cdot \text{OPT}.$$

For the same reasons we get $f_{\text{dur}}(M(T)) \leq O(\log^{2p} m) \cdot \text{OPT}$, and thus the solution $x^{(1)}, \ldots, x^{(T)}$ is $O(\log^{2p} m)$-competitive for the instance $\mathcal{I}_{\text{onGAP}}$: 

$$f_{\text{dur}}(\Lambda(T)) + f_{\text{dur}}(M(T)) \leq O(\log^{2p} m) \cdot \text{OPT}.$$

This concludes the proof of Theorem 1.4. \qed
5 Applications to Bandit Algorithms

In this section we use gradient stable approximations of norms to design bandit algorithms.

5.1 Approximate Converse to Jensen’s Inequality

Along with smoothness, a key technical observation to design our bandit algorithms is that the increase of $\Psi$ can be bounded in terms of a surrogate cost, which is defined based on its gradient. Note that convexity of $\Psi$ immediately implies that $\langle \nabla \Psi(\Lambda_{(t-1)}), y^{(t)} \rangle \leq \Psi(\Lambda_{(t-1)}) - \Psi(\Lambda_{(t-1)})$. However, gradient stability also implies the following reverse bound.

**Lemma 5.1.** Consider any sequence of non-negative vectors $y^{(1)}, \ldots, y^{(T)} \in [0,1]^d$. Furthermore, let $\Lambda_{(t)} = \sum_{s=1}^{t} y^{(s)}$ be the sum of the first $t$ vectors. For any norm $\| \cdot \|$ that admits a $\delta$-gradient-stable approximation with error $(\alpha, \gamma)$, we have for every $\epsilon > 0$ that

$$
\sum_{t=1}^{T} \langle \nabla \Psi_{e}(\Lambda_{(t-1)}), y^{(t)} \rangle \geq \exp(-\epsilon \|1\| - \delta) \left( \Psi_{e}(\Lambda_{(T)}) - \Psi_{e}(\Lambda_{0}) - \Psi_{e}(1) \right).
$$

**Proof.** Define $\tilde{\Lambda}_{(t)} = \min\{0, \Lambda_{(t)} - 1\}$ and $\tilde{y}^{(t)} = \tilde{\Lambda}_{(t)} - \tilde{\Lambda}_{(t-1)}$. Notice that $y^{(t)} \geq \tilde{y}^{(t)}$ component-wise for all $t$. This also implies that $\tilde{y}_{i}^{(t)} \leq 1$ for all $i$, so by monotonicity $\|\tau \tilde{y}^{(t)}\| \leq \|1\|$ for all $\tau \in [0,1]$, which also implies $\nabla \Psi_{e}(\tilde{\Lambda}_{(t-1)} + \tau \tilde{y}^{(t)}) \geq \exp(-\epsilon \|1\| - \delta) \nabla \Psi_{e}(\tilde{\Lambda}_{(t-1)})$ by gradient stability. This implies

$$
\langle \nabla \Psi_{e}(\Lambda_{(t-1)}), y^{(t)} \rangle \geq \int_{0}^{1} \langle \nabla \Psi_{e}(\tilde{\Lambda}_{(t-1)} + \tau \tilde{y}^{(t)}), \tilde{y}^{(t)} \rangle \, d\tau \geq \exp(-\epsilon \|1\| - \delta) \int_{0}^{1} \langle \nabla \Psi_{e}(\tilde{\Lambda}_{(t-1)}), \tilde{y}^{(t)} \rangle \, d\tau = \exp(-\epsilon \|1\| - \delta) \left( \Psi_{e}(\tilde{\Lambda}_{(t)}) - \Psi_{e}(\tilde{\Lambda}_{(t-1)}) \right).
$$

By a telescoping sum, we get

$$
\sum_{t=1}^{T} \langle \nabla \Psi_{e}(\Lambda_{(t-1)}), y^{(t)} \rangle \geq \exp(-\epsilon \|1\| - \delta) \left( \Psi_{e}(\tilde{\Lambda}_{(T)}) - \Psi_{e}(\tilde{\Lambda}_{(0)}) \right).
$$

Furthermore, by subadditivity $\Psi_{e}(\tilde{\Lambda}_{(T)}) \geq \Psi_{e}(\tilde{\Lambda}_{(T)}) - \Psi_{e}(\tilde{\Lambda}_{(T)}) \geq \Psi_{e}(\tilde{\Lambda}_{(T)}) - \Psi_{e}(1)$ and by definition $\Psi_{e}(\tilde{\Lambda}_{(0)}) = \Psi_{e}(0) = \Psi_{e}(\Lambda_{0})$. This implies the lemma. □

We now have the tools for our bandit applications.

5.2 Bandits with Knapsacks

We will first consider the following Bandits with Knapsacks (BwK) problem for adversarial arrivals. Here, an algorithm chooses one of $n$ actions in every time step. Each action gives a reward and incurs a vector load. The process stops when a norm of the sum of vector loads exceeds some given budget $B$, or when the time horizon $T$ is reached.

In more detail, an adversary initially chooses $T$ reward vectors $r^{(1)}, \ldots, r^{(T)} \in [0,1]^d$ and $T$ load matrices $C^{(1)}, \ldots, C^{(T)} \in [0,1]^{d \times n}$, which are unknown to the algorithm. In time step $t$, the algorithm chooses an action, where the $i$-th action gives reward $r^{(t)}_{i} \in [0,1]$ and incurs a vector load of $C^{(t)} e_{i} \in [0,1]^d$, both of which are unknown before playing the action. Importantly, there is also a null action, not causing any reward nor load. We let $x^{(t)} \in \{e_{1}, \ldots, e_{n}\}$ denote the vector...
indicating which action is chosen in step $t$. As soon as the algorithm reaches a point such that $\| \sum_{t' < t} C^{(t')} x^{(t')} \| > B$, only the null action can be chosen. After the algorithm has chosen the action, it only gets to know $\langle r^{(t)}, x^{(t)} \rangle$ and $C^{(t)} \cdot x^{(t)}$ (bandit feedback).

The benchmark for BWK is defined as follows: For any fractional choice $x^* \in \Delta_n := \{ x \in [0, 1]^n \mid \sum_{i=1}^n x_i = 1 \}$ of the $n$ actions, we let $\tau^* := \min \{ t \mid \| \sum_{i=1}^t C^{(i)} \cdot x^* \| > B \}$ be the time step at which $x^*$ would run out of budget, or $\tau^* = T$ if there is budget left. Then the optimum $\text{OPT}_{\text{BWK}}$ is defined the maximum over all $\sum_{i=1}^{\tau^*} \langle r^{(i)}, x^* \rangle$.

**Theorem 1.5.** Consider the Bandits with Knapsacks problem for adversarial arrivals with a norm $\| \cdot \|$. Suppose $\| \cdot \|$ admits a $\frac{1}{4}$-gradient-stable approximation with error $(\alpha, \gamma)$. Furthermore, let $B \geq 4 \cdot (\alpha + \gamma) \cdot \| 1 \|$. Then, there exists an algorithm that takes $\text{OPT}_{\text{BWK}}$ as its input and obtains reward at least

$$\Omega \left( \frac{1}{\alpha + \gamma} \text{OPT}_{\text{BWK}} \right) - O \left( \frac{\text{OPT}_{\text{BWK}} \cdot \| 1 \|}{(\alpha + \gamma) \cdot B} \right) \cdot \text{REGRET},$$

with probability $1 - p$, where $\text{REGRET} = O(\sqrt{Tn \log(n/p)})$ and $p \in [0, 1]$ is a parameter. Moreover, this algorithm is efficient given gradient oracle access to $\frac{1}{4}$-gradient stable approximation of $\| \cdot \|$.

The assumption that $\text{OPT}_{\text{BWK}}$ is known can be removed in this theorem at a further multiplicative loss of $O(\log T)$, which is known to be unavoidable [ISSS19].

**Algorithm 2** Bandits with Knapsacks

**input:** Estimate $\text{OPT}_{\text{BWK}}$

- Let $\Psi = \Psi_\epsilon$ be a $\frac{1}{4}$-gradient-stable approximation of $\| \cdot \|$ with error $(\alpha, \gamma)$ at scale $\epsilon = \frac{4(\alpha + \gamma)}{B}$.

- for each time $t$
  - Adopt choice of action $x^{(t)}$ from bandits algorithm (e.g. Exp3.P).
  - Return $\langle R^{(t)}, x^{(t)} \rangle$ to bandits algorithm as the reward as defined by Equations (9) and (10).
- end for

Our algorithm, Algorithm 2, generalizes the one in [KS20]. It uses a $\frac{1}{4}$-gradient-stable approximation $\Psi = \Psi_\epsilon$ of $\| \cdot \|$ with error $(\alpha, \gamma)$ for $\epsilon = \frac{4(\alpha + \gamma)}{B}$. Note that this choice of $\epsilon$ is feasible because $\alpha$ and $\gamma$ do not depend on $\epsilon$. Based on this norm approximation, it defines a surrogate game, which can be viewed as a Lagrangian relaxation of the original problem. In this surrogate game it applies a classic no-regret bandits algorithm, which is allowed to choose actions freely without any constraints, such as Exp3.P [ACFS02]. Generally, we do not need any property besides a bound on the regret with probability $1 - p$ against an adaptive adversary for an unknown time horizon.

Specifically, the $i$-th action’s surrogate reward in time step $t$ is defined as

$$R_i^{(t)} = \begin{cases} r^{(t)} e_i - \lambda \cdot \langle \nabla \Psi(\Lambda^{(t-1)}), C^{(t)} \cdot e_i \rangle & \text{if } \| \Lambda^{(t-1)} \| \leq B \\ 0 & \text{otherwise,} \end{cases}$$

where $r^{(t)}$ is the reward vector from BwK problem, $\Lambda^{(t-1)} = \sum_{s=1}^{t-1} C^{(s)} \cdot x^{(s)}$ is the load after time $t - 1$, and

$$\lambda := \frac{\sum_{i=1}^{\tau^*} \langle r^{(t)}, x^* \rangle}{6e^\gamma(\alpha + \gamma)B} = \frac{\text{OPT}_{\text{BWK}}}{6e^\gamma(\alpha + \gamma)B},$$

Equations (9) and (10).
Note that in order to define \( \lambda \), our algorithm has to know the value of \( \OPT_{\text{BwK}} \), which we assumed to be part of the input.

In order to bound the regret an algorithm could guarantee in this surrogate game, observe that by Lemma A.5, we have \( (C(t)e_i, \nabla \Psi(\Lambda^{(t-1)})) \leq \alpha \|1\| \) and consequently \( |R^{(t)}_1| \leq \rho \) for \( \rho := 1 + \lambda \cdot \alpha \cdot \|1\| \).

Therefore, we can we can guarantee that

\[
\sum_{t=1}^{\tau^*} \langle R^{(t)}, x(t) \rangle \leq \sum_{t=1}^{\tau^*} \langle R^{(t)}, x^* \rangle + \rho \cdot \text{REGRET}_{\tau^*}, \tag{11}
\]

where \( \text{REGRET}_{\tau^*} \) would be the regret of the algorithm against an adaptive adversary choosing losses in \([-1, 1]\). Note that \( \tau^* \) is not known to the algorithm. However, it is a fixed number not depending on the choices of the algorithm. So it is sufficient to apply any algorithm operating with an unknown time horizon. Here, \( \text{Exp3.P} \) [ACFS02] would give us \( \text{REGRET}_{\tau^*} = O(\sqrt{\tau^* n \log(n/p)}) \) with probability \( 1 - p \).

Observe that the algorithm only needs one gradient query of \( \Psi(\cdot) \) per time step. Now, it remains to show that the algorithm obtains the claimed reward.

**Proposition 5.2.** Whenever Equation (11) is fulfilled and \( B \geq 4 \cdot (\alpha + \gamma) \cdot \|1\| \), the reward obtained by actions \( x^{(1)}, \ldots, x^{(T)} \) in the BwK problem fulfills

\[
\sum_{t=1}^{\tau} \langle r(t), x(t) \rangle \geq \Omega\left(\frac{1}{\alpha + \gamma} \OPT_{\text{BwK}}\right) - O\left(\frac{\OPT_{\text{BwK}} \cdot \|1\|}{(\alpha + \gamma) \cdot B}\right) \cdot \text{REGRET}_{\tau^*}.
\]

**Proof.** We distinguish the analysis in two cases: Either the algorithm stops before time \( \tau^* \) (recall, this is time at which \( x^* \) runs out of budget, otherwise \( \tau^* = T \)) or it stays within budget until \( \tau^* \).

**Case 1:** \( \Psi(\Lambda^{(\tau^*)}) > B \). As \( \Psi(\Lambda^{(\tau^*)}) > B \) implies that \( \|\Lambda^{(\tau^*)}\| > B \), there is some time \( \tau \leq \tau^* \) at which \( \|\Lambda^{(\tau)}\| > B \) for the first time. This means that \( \tau \) is the last round before the algorithm stops and \( R^{(t)} = 0 \) for all \( t > \tau \). We compare to always playing the null action, which has surrogate reward 0. By Equation (11), in the surrogate game, we therefore get

\[
\sum_{t=1}^{\tau} \langle R^{(t)}, x(t) \rangle = \sum_{t=1}^{\tau^*} \langle R^{(t)}, x(t) \rangle \geq 0 - \rho \cdot \text{REGRET}_{\tau^*}.
\]

Using the definition of \( R^{(t)} \), this implies

\[
\sum_{t=1}^{\tau} \langle r(t), x(t) \rangle \geq \lambda \sum_{t=1}^{\tau} \langle \nabla \Psi(\Lambda^{(t-1)}), C(t)x(t) \rangle - \rho \cdot \text{REGRET}_{\tau^*}.
\]

Since \( \Psi \) approximates the norm, we have \( \Psi(\Lambda^{(\tau)}) - \Psi(0) - \Psi(1) \geq B - (\alpha \|1\| + \frac{\alpha}{2}) - (\alpha \cdot 0 + \frac{\alpha}{2}) \geq \frac{B}{4} \), where in the last step we use the definition of \( \epsilon \) and that \( B \geq 4(\alpha + \gamma)\|1\| \).

So, by Lemma 5.1,

\[
\sum_{t=1}^{\tau} \langle \nabla \Psi(\Lambda^{(t-1)}), C(t)x(t) \rangle \geq \exp(-\epsilon \|1\| - \delta)(\Psi(\Lambda^{(\tau)}) - \Psi(0) - \Psi(1))
\]

25
\[ \geq \frac{B}{4} \exp(-\epsilon \|1\| - \delta) \geq \frac{B}{4} \exp(-5). \]

This implies
\[
\sum_{t=1}^{\tau^*} \langle r(t), x(t) \rangle \geq \lambda \cdot \frac{B}{4e^{\beta}} - \rho \cdot \text{REGRET}_{\tau^*} \geq \frac{\text{OPT}_{BwK}}{24e^{10(\alpha + \gamma)}} - \rho \cdot \text{REGRET}_{\tau^*}.
\]

**Case 2:** \( \Psi(\Lambda^{(\tau^*)}) \leq B \). In this case, \( \|\Lambda(t)\| \leq \Psi(\Lambda^{(\tau^*)}) \leq B \) for all \( t \leq \tau^* \). As an auxiliary point of comparison, we now use a scaled-down version of the optimal choice, namely, \( x' := \frac{1}{\alpha + \gamma} x^* \).

Note that also \( x' \) is a feasible fractional solution because we assume that the rest is a null action.

By Equation (11), we get \( \sum_{t=1}^{\tau^*} \langle \mathcal{R}(t), x(t) \rangle \geq \sum_{t=1}^{\tau^*} \langle \mathcal{R}(t), x' \rangle - \rho \cdot \text{REGRET}_{\tau^*} \), and therefore by the definition of \( \mathcal{R}(t) \),
\[
\sum_{t=1}^{\tau^*} \langle r(t), x(t) \rangle \geq \sum_{t=1}^{\tau^*} \left( \langle r(t), x' \rangle - \lambda \cdot \langle \nabla \Psi(\Lambda^{(t-1)}), C(t)x' \rangle + \lambda \cdot \langle \nabla \Psi(\Lambda^{(t-1)}), C(t)x(t) \rangle \right) - \rho \cdot \text{REGRET}_{\tau^*} \\
\geq \frac{\text{OPT}_{BwK}}{\alpha + \gamma} - \lambda \left( \sum_{t=1}^{\tau^*} \langle \nabla \Psi(\Lambda^{(t-1)}), C(t)x' \rangle \right) - \rho \cdot \text{REGRET}_{\tau^*},
\]

where the last step uses that \( \sum_{t=1}^{\tau^*} \langle r(t), x' \rangle = \frac{\text{OPT}_{BwK}}{\alpha + \gamma} \) and that all entries in \( C(t) \) as well as the gradients of \( \Psi \) are non-negative.

Define \( y^*_t = \frac{1}{\alpha + \gamma} C(t)x' \) and \( y^t = C(t)x(t) \). By convexity of \( \Psi \), we have
\[
\sum_{t=1}^{\tau^*} \langle \nabla \Psi(\Lambda^{(t-1)}), y^*_t \rangle \leq \sum_{t=1}^{\tau^*} \left( \Psi(\Lambda^{(t-1)}) + y^*_t - \Psi(\Lambda^{(t-1)}) \right).
\]

Now, we use Lemma 3.2, replacing \( T \) by \( \tau^* \) in the statement, and the fact that \( \Psi(\Lambda^{(\tau^*)}) \leq \alpha - \frac{B}{\alpha + \gamma} + \frac{\gamma}{\epsilon} \leq 2B \) and that \( \Psi(\Lambda^{(\tau^*)}) \leq B \). This gives us that
\[
\sum_{t=1}^{\tau^*} \left( \Psi(\Lambda^{(t-1)}) + y^*_t - \Psi(\Lambda^{(t-1)}) \right) \leq \exp \left( \frac{\epsilon B}{\alpha + \gamma} + \delta \right) 3B \leq 3e^5 B.
\]

So, we get
\[
\sum_{t=1}^{\tau^*} \langle r(t), x(t) \rangle \geq \frac{\text{OPT}_{BwK}}{\alpha + \gamma} - \lambda \cdot 3e^5 B - \rho \cdot \text{REGRET}_{\tau^*} = \frac{\text{OPT}_{BwK}}{6e^{10}(\alpha + \gamma)} - \rho \cdot \text{REGRET}_{\tau^*}. \]

**Remark 5.3.** If the algorithm gets to know the entire vector \( r(t) \) and the entire matrix \( C(t) \) after step \( t \) the bound in Theorem 1.5 can be improved to \( \text{REGRET} = O(\sqrt{T \log n}) \), namely by using HEDGE \([FS99]\).
5.3 Bandits with Vector Costs

In this section we consider the Bandits with Vector Costs (BwVC) problem, which is a natural generalization of the classical bandits problem when the actions incur vector costs. Here, in each time step \( t \), an algorithm can choose one of \( n \) actions (“experts”). In each time step, each action has a \( d \)-dimensional cost vector associated to it and the goal is to minimize a norm on the sum of cost vectors. That is, an adversary defines \( T \) matrices \( C^{(1)}, \ldots, C^{(T)} \in [0, 1]^{d \times n} \) and the algorithm chooses vectors \( x^{(1)}, \ldots, x^{(T)} \in \{0, 1\}^n \), each containing exactly one entry that is 1, attempting to minimize \( \| \sum_{t=1}^{T} C^{(t)} \cdot x^{(t)} \| \). The key difference to generalized load balancing is that the algorithm only gets to know \( \langle C^{(t)}, x^{(t)} \rangle \) only after it has chosen \( x^{(t)} \).

The benchmark is the best fractional choice of actions. That is, we consider \( x^* \in \Delta_n := \{ x \in [0, 1]^n \mid \sum_{i=1}^{n} x_i = 1 \} \) so that \( \| \sum_{t=1}^{T} C^{(t)} \cdot x^* \| \) is minimized.

We define \( \Lambda^{(t)} = \sum_{s=1}^{t} C^{(t)} \cdot x^{(t)} \) as sum of the vector costs incurs in the first \( t \) steps and \( \Lambda^*_t = \sum_{s=1}^{t} C^{(t)} \cdot x^* \) the respective quantity for the benchmark solution.

**Theorem 1.6.** Consider the problem Bandits with Vector Costs with a norm \( \| \cdot \| \). If \( \| \cdot \| \) admits a \( 1/4 \)-gradient-stable approximations with error \( (\alpha, \gamma) \), then there exists an algorithm that guarantees \( \| \Lambda^{(T)} \| = O((\alpha + \gamma) \cdot \| \Lambda^*_T \| + \alpha \cdot \| 1 \| \cdot \text{REGRET}) \) with probability \( 1 - p \), where \( \text{REGRET} = O(\sqrt{Tn \log(n/p)}) \) and \( p \in [0, 1] \) is a parameter. Moreover, this algorithm is efficient given gradient oracle access to this \( 1/4 \)-gradient-stable approximation.

Our algorithm for BwVC reduces the problem to classic online learning with a scalar cost. It then adopts the action chosen by this algorithm. To this end it uses a gradient-stable approximation of the norm as follows. In each step, we define a surrogate linear cost function by defining \( c^{(t)} = (\nabla\Psi(\Lambda^{(t-1)}))^\text{transposed} \cdot C^{(t)} \). That is, the surrogate scalar cost of \( x \) is \( \langle c^{(t)}, x \rangle = \langle \nabla\Psi(\Lambda^{(t-1)}), C^{(t)} \cdot x^{(t)} \rangle \). So, we approximate \( \Psi \) by its gradient at the current load \( \Lambda^{(t-1)} \).

By Lemma A.5 and because \( C^{(t)} \in [0, 1]^{d \times n} \), we always have \( c^{(t)} \in [0, \rho] \) for \( \rho := \alpha \| 1 \| \). Therefore, applying any no-regret learning algorithm for a scalar cost problem. Using EXP3.P [ACFS02] we are guaranteed that with probability \( 1 - p \)

\[
\sum_{t=1}^{T} \langle c^{(t)}, x^{(t)} \rangle \leq \sum_{t=1}^{T} \langle c^{(t)}, x^* \rangle + \rho \cdot \text{REGRET},
\]

(12)

where \( \text{REGRET} = O(\sqrt{Tn \log(n/p)}) \).

Observe that the algorithm only needs one gradient query of \( \Psi \) per time step. Now it only remains to prove the following proposition in order to show the theorem.

**Proposition 5.4.** Whenever (12) is satisfied, \( \| \Lambda^{(T)} \| = O((\alpha + \gamma) \cdot \| \Lambda^*_T \| + \rho \cdot \text{REGRET}) \).

**Proof.** Letting \( y^{(t)}_* = C^{(t)} \cdot x^* \), we have by convexity of \( \Psi \)

\[
\langle \nabla\Psi(\Lambda^{(t-1)}), y^{(t)}_* \rangle \leq \Psi(\Lambda^{(t-1)} + y^{(t)}_*) - \Psi(\Lambda^{(t-1)});
\]

so also by the definition of \( c^{(t)} \) and by Lemma 3.2

\[
\sum_{t=1}^{T} \langle c^{(t)}, x^* \rangle \leq \sum_{t=1}^{T} \Psi(\Lambda^{(t-1)} + y^{(t)}_*) - \Psi(\Lambda^{(t-1)}) \leq \zeta \Psi(\Lambda^*_T) + (\zeta - 1) \left( \Psi(\Lambda^{(T)}) - \Psi(\Lambda^{(0)}) \right),
\]

27
where \( \zeta = \exp \left( \epsilon \| \Lambda_*^{(T)} \| + \delta \right) \).

Combining this bound with Lemma 5.1 and the regret property, we get

\[
\Psi(\Lambda^{(T)}) - \Psi(\Lambda^{(0)}) - \Psi(1) \leq \exp(\epsilon \| 1 \| + \delta) \sum_{t=1}^{T} \langle c(t), x(t) \rangle
\leq \exp(\epsilon \| 1 \| + \delta) \left( \rho \cdot \text{Regret} + \sum_{t=1}^{T} \langle c(t), x^* \rangle \right)
\leq \exp(\epsilon \| 1 \| + \delta) \left( \rho \cdot \text{Regret} + \zeta \Psi(\Lambda_*^{(T)}) + (\zeta - 1) \left( \Psi(\Lambda^{(T)}) - \Psi(\Lambda^{(0)}) \right) \right).
\]

This implies

\[
(2 - \exp(\epsilon \| 1 \| + \epsilon \| \Lambda_*^{(T)} \| + 2\delta)) \left( \Psi(\Lambda^{(T)}) - \Psi(\Lambda^{(0)}) \right)
\leq \exp(\epsilon \| 1 \| + \epsilon \| \Lambda_*^{(T)} \| + 2\delta) \left( \Psi(\Lambda_*^{(T)}) + \rho \cdot \text{Regret} \right) + \Psi(1).
\]

For \( \epsilon := 1/(4 \cdot \| \Lambda_*^{(T)} \| + 4\| 1 \|) \), \( \delta := 1/8 \), we have \( \exp(\epsilon \| \Lambda_*^{(T)} \| + \epsilon \| 1 \| + 2\delta) \leq \exp(1/4 + 1/4) \leq \frac{9}{5} \).

And thus,

\[
\frac{1}{5} \left( \Psi(\Lambda^{(T)}) - \Psi(\Lambda^{(0)}) \right) \leq \frac{9}{5} \left( \Psi(\Lambda_*^{(T)}) + \rho \cdot \text{Regret} \right) + \Psi(1),
\]

or equivalently, \( \Psi(\Lambda^{(T)}) \leq 9 \cdot \Psi(\Lambda_*^{(T)}) + \Psi(\Lambda^{(0)}) + 5 \cdot \Psi(1) + 9 \cdot \rho \cdot \text{Regret} \).

By the approximation properties of \( \Psi \), we get

\[
\| \Lambda^{(T)} \| \leq 9 \cdot \alpha \cdot \| \Lambda_*^{(T)} \| + 5 \cdot \alpha \cdot \| 1 \| + 15 \cdot \frac{\gamma}{\epsilon} + 9 \cdot \rho \cdot \text{Regret}.
\]

Remark 5.5. If the algorithm gets to know the entire matrix \( C^{(t)} \) after step \( t \), the bound in Theorem 1.6 can be improved to \( \text{Regret} = O(\sqrt{T \log n}) \) by using Hedge [FS99].

6 Conclusions and Further Directions

Our gradient-stable norm approximation gives \( O(\log^2 d) \)-competitive algorithms for several online and bandit problems, in particular for Online Generalized Load Balancing and Bandits with Knapsacks for symmetric norms. There are still many open problems and future directions. For example, we only know \( \Omega(\log d) \) lower bounds for both these problems, which leaves open what are the tight competitive ratios. Another interesting direction is to consider other online problems with (symmetric) norm objectives, e.g. Online Set Cover [ABC+16, SN20], and obtain polylog \( d \)-competitive algorithms. Besides our approach via gradient-stable norm approximation, it would be also interesting to extend other techniques such as online primal-dual (see [BN09]) to online problems with symmetric norms. Finally, it is an intriguing question to better understand arbitrary monotone norms and whether they admit good gradient-stable approximations.

Acknowledgments

We are grateful to the anonymous reviewers of SODA 2023 for their helpful comments.
A Further Related Work and Missing Proofs

A.1 Further Related Work

Other notions of “smoothness”. “Smoothness” of functions (in a broad sense) has been recognized to play a very important role on an multitude of settings, and a comprehensive survey about the subject is impossible. Nonetheless, we briefly discuss notions that are most closely related to the gradient-stability property

\[ \nabla \Psi(x + y) \geq e^{-\epsilon \|y\| - \delta} \cdot \nabla \Psi(x), \quad \forall x, y \in \mathbb{R}^d. \]  

(13)

The most classic such notion is that of a function \( \Psi \) having Lipschitz gradient, namely

\[ \|\nabla \Psi(x + y) - \nabla \Psi(x)\| \leq L \cdot \|y\| \]  

for some norms \( \| \cdot \|_{(1)} \), \( \| \cdot \|_{(2)} \) and a constant \( L \). A main difference between this and gradient-stability (13) is that in the latter we have a multiplicative bound between \( \nabla \Psi(x + y) \) and \( \nabla \Psi(x) \); this seems to be critical for the “error” between these gradients not to accumulate as we add over all the time steps in the analysis of online problems.

Another classic notion from the theory of Banach spaces is the so-called uniform smoothness of norms [LT13]. However, if \( \| \cdot \| \) is uniformly smooth (and twice differentiable, for simplicity) we only seem to obtain a bound of the form \( y^\top (\nabla^2 \|x\|) y \leq C \|y\|^2 \) for some constant \( C \) and all \( x, y \), which again only means that \( \| \cdot \| \) has Lipschitz gradients.

Notions of “smoothness” of the so-called baseline potential (which translate to stability of decisions) have also played a fundamental role in determining the regret in online learning algorithms [ALST14, ALT15, Mol17, KS20, Mol20]. Indeed, in the standard algorithm FTRL algorithm the addition of the strongly convex regularizer translates to making the baseline potential strongly smooth, or equivalently, having Lipschitz gradients. In addition, [Mol17] also explicitly used the multiplicative smoothing condition

\[ \nabla \Psi(x + y) \in e^{\pm \epsilon \|y\|_\infty} \cdot \nabla \Psi(x), \quad \forall x, y \in \mathbb{R}_+^d, \]  

(14)

which is a special case of gradient-stability where the length of the change vector \( y \) is always measured in \( \ell_\infty \)-norm (this is implicitly present in the original FTPL analysis [KV05]; see also [ALT15] for a related notion). As discussed in Section 1.5, such approximation to a norm \( \| \cdot \| \) can be obtained by setting \( \Psi(x) = \mathbb{E}_\nu \|x + \nu\| \) where each coordinate of \( \nu \) is an independent Exponential random variable with mean \( 1/\epsilon \). However, also as discussed there, all known approximations satisfying (14) have a large error \( \Psi(0) \gg \|0\| \) and yield online algorithms with very poor approximation guarantees. This weakness can be traced to the fact the guarantee in (14) is not tailored to the specific norm \( \| \cdot \| \).

Finally, [GSS21] introduced the closely related notion of one-sided smoothness in the context of DR-submodular optimization. Adapted to convex functions, this definition requires a (twice differentiable) function to satisfy \( y^\top \nabla^2 \Psi(x) y \geq -C \frac{\|y\|_1}{\|x\|_1} \langle y, \nabla \Psi(x) \rangle \), noting that the bound is based specifically on the \( \ell_1 \)-norm. In contrast, gradient-stability (with \( \delta = 0 \)) implies \( y^\top \nabla^2 \Psi(x) y \geq -\epsilon \|y\|_1 \langle y, \nabla \Psi(x) \rangle \), with the bound depending on the norm \( \| \cdot \| \); as in the previous paragraph, this tailoring to the given norm \( \| \cdot \| \) seems to be crucial for obtaining low-error approximations. In addition, [GSS21] does not discuss how to approximate a given function/norm by a one-sided smooth function.
**Other related results in online algorithms.** There are other results on online algorithm for problems with convex objectives that are related to our present work. However most of them assume that the objective function $f$ has monotone gradients, namely $\nabla f(x') \geq \nabla f(x)$ for all $x' \geq x \geq 0$ [BGMS11, HK15, ABC+16, EFM16, GMM18, Mol21]. Notice that monotone norms are far from satisfying monotone gradients property, e.g., $\nabla \| (1, 0) \|_2 = (1, 0)$ but $\nabla \| (1, 1) \|_2 = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$.

The only exception to gradient monotonicity that we know is [SN20], where the authors consider a version of the Online Set Cover problem where the objective function is a sum of $\ell_p$ norms. However, in this case we have the crucial property that the function $\| \cdot \|_p$ does have monotone gradients, a fact that is crucially exploited in their analysis.

Also notice that gradient-stability can be thought of as an appropriately defined relaxation of gradient monotonicity that is still enough to yield algorithms with good approximation guarantees.

### A.2 Softmax and $\ell_p$ Norms

Recall that $\nabla_i f$ denotes the $i$-th partial derivative of a function $f$.

**Softmax function.** A standard “smooth” approximation of the $\ell_\infty$-norm is the softmax function $SM_\epsilon(x) := \frac{1}{\epsilon} \ln \left( \sum_i e^{\epsilon x_i} \right)$. The next lemma shows that the softmax actually provides a gradient-stable approximation.

**Lemma A.1.** The softmax function provides a 0-gradient-stable approximation of $\| \cdot \|_\infty$ (in $d$-dimensions) with error $(1, \ln d)$.

**Proof.** Fix $\epsilon > 0$ throughout. First, it is clear that the function $SM_\epsilon$ is monotone, and it is well-known that it is also convex. It is also subadditive:

$$SM_\epsilon(x + y) = \frac{1}{\epsilon} \ln \left( \sum_i e^{\epsilon x_i} e^{\epsilon y_i} \right) \leq \frac{1}{\epsilon} \ln \left( \left( \sum_i e^{\epsilon x_i} \right) \cdot \left( \sum_i e^{\epsilon y_i} \right) \right) = SM_\epsilon(x) + SM_\epsilon(y).$$

For the approximation error, we have $SM_\epsilon(x) = \frac{1}{\epsilon} \ln \left( \sum_i e^{\epsilon x_i} \right) \geq \frac{1}{\epsilon} \ln \left( e^{\|x\|_\infty} \right) = \|x\|_\infty$. In the other direction, we have $SM_\epsilon(x) = \frac{1}{\epsilon} \ln \left( \sum_i e^{\epsilon x_i} \right) \leq \frac{1}{\epsilon} \ln \left( d \cdot e^{\|x\|_\infty} \right) = \|x\|_\infty + \ln d$.

Next, we compute the gradients: for every $i \in [d]$

$$\nabla_i SM_\epsilon(x) = \frac{e^{\epsilon x_i}}{\sum_j e^{\epsilon x_j}} \quad \text{and} \quad \nabla_i SM_\epsilon(x + y) = \frac{e^{\epsilon x_i + \epsilon y_i}}{\sum_j e^{\epsilon x_j + \epsilon y_j}} \geq \frac{e^{\epsilon x_i}}{\sum_j e^{\epsilon x_j}} \cdot e^{\epsilon \|y\|_\infty}.$$

Thus, we get

$$\nabla_i SM_\epsilon(x + y) \geq \exp(-\epsilon \|y\|_\infty) \cdot \frac{e^{\epsilon x_i}}{\sum_j e^{\epsilon x_j}} = \exp(-\epsilon \|y\|_\infty) \cdot \nabla_i SM_\epsilon(x).$$

Since these bounds hold for every $\epsilon > 0$, it proves the desired result. \qed
\( \ell_p \text{ Norms.} \) The more general \( \ell_p \)-norms with \( p \in [1, \infty) \) also admit gradient-stable approximations.

**Lemma A.2.** For every \( p \in [1, \infty) \), the \( \ell_p \)-norm admits a 0-gradient-stable approximation with error \( (1, \min\{p - 1, \ln d\}) \).

\[ \text{Proof.} \] Fix \( \epsilon > 0 \) and define the approximation \( \Psi(x) = \|x + c1\|_p - c \), where \( c = \frac{p - 1}{\epsilon \|1\|_p} \).

We see that \( \Psi \) is monotone, convex (since the \( \ell_p \)-norm is convex); it is also subadditive, since by triangle inequality

\[ \Psi(x + y) = \|x + y + c1\|_p \leq \|x + c1\|_p + \|y\|_p \leq \Psi(x) + \Psi(y). \]

For the other desired properties, we first see that \( \|x\|_p \leq \Psi(x) \). This is because by convexity \( \|x + c1\|_p - \|x\|_p \geq \langle \nabla \|x\|_p, c1 \rangle = c \|\nabla \|x\|_p\|_1 \geq c \). Furthermore, by triangle inequality \( \Psi(x) \leq \|x\|_p + \|c1\|_p - c \) and observe that \( \|c1\|_p - c = \frac{(p-1)(\|1\|_p-1)}{c\|1\|_p} \leq \frac{1}{\epsilon} \min\{p - 1, \ln d\} \). So it has approximation error \( (1, \min\{p - 1, \ln d\}) \). Next, we compute the gradients: for all \( x, y \in \mathbb{R}_+^d \)

\[ \nabla_i \Psi(x) = \frac{(x_i + c)^{p-1}}{\|x + c1\|_p^{p-1}} \quad \text{and} \quad \nabla_i \Psi(x + y) = \frac{(x_i + y_i + c)^{p-1}}{\|x + y + c1\|_p^{p-1}}. \]

Observe that by triangle inequality,

\[ \|x + y + c1\|_p \leq \|x + c1\|_p + \|y\|_p = \|x + c1\|_p \left( 1 + \frac{\|y\|_p}{\|x + c1\|_p} \right) \leq \|x + c1\|_p \left( 1 + \frac{\|y\|_p}{c\|1\|_p} \right) \leq \exp \left( \frac{\|y\|_p}{c\|1\|_p} \right) \|x + c1\|_p. \]

Therefore,

\[ \nabla_i \Psi(x + y) = \frac{(x_i + y_i + c)^{p-1}}{\|x + y + c1\|_p^{p-1}} \geq \exp \left( -\frac{(p-1)\|y\|_p}{c\|1\|_p} \right) \frac{(x_i + c)^{p-1}}{\|x + c1\|_p^{p-1}} = \exp(-\|y\|_p) \nabla_i \Psi(x), \]

which proves its gradient-stability. \( \square \)

### A.3 Smoothing of Top-k Norm in Polynomial Time

**Theorem 2.3.** For every \( \delta, \rho \in (0, 1) \), there is an algorithm that with probability at least \( 1 - \rho \) constructs an approximation of the Top-k norm in \( \mathbb{R}_+^d \) with error

\[ \left( \exp(\delta) \left( 1 + \frac{4}{\delta} \right), \exp(2\delta) \frac{4Hd}{\delta} \right) \]

that is differentiable almost-everywhere and wherever it is differentiable it satisfies \( \delta \)-gradient-stability. Moreover, one can perform value and gradient oracle calls in time polynomial in \( \frac{d}{\rho} \).

Recall that the existential construction for a \( \delta' \)-gradient stable approximation in Section 2.1 uses

\[ f_e(x) := E_{K, \nu} \|x + \nu\|_{\text{Top-K}}, \quad \text{where} \]

\[ (15) \]
1. \( \nu_i \) is an independent Exponential random variable with rate \( \lambda = \epsilon k \) for every \( i \in [d] \).

2. \( K \) follows the geometric distribution starting at 1 with parameter \( p = 1 - q \) for \( q = 1 - \frac{\theta'}{\kappa} \), i.e., \( \Pr(K = i) = q^{i-1}(1 - q) \) for \( i \geq 1 \) and \( \mathbb{E}[K] = \frac{k}{\theta'} \).

3. For \( i > d \), we define \( \|x\|_{\text{Top-}i} = \|x\|_{\text{Top-}d} \).

To obtain a \( \delta \)-gradient stable approximation in polynomial time, we approximate \( \nu \) by samples and set \( \delta' = \frac{1}{4} \delta \). That is, for an \( s \), which is appropriately chosen as a polynomial in \( d \rho \) to be defined later, let \( V^1, \ldots, V^s \in \mathbb{R}^d \) be independent RV's where each coordinate \( V^\ell_i \) is an independent Exponential random variable with parameter \( \lambda = \epsilon k \). Then instead of \( f \) we use

\[
\hat{f}(x) = \mathbb{E}_K \frac{1}{s} \sum_{\ell=1}^s \|x + V^\ell\|_{\text{Top-}K} + \frac{2\Delta}{\delta''} \cdot \|x\|_1,
\]

where \( \delta'' = \frac{1}{2} \delta \). Since the random variable \( K \) only has \( d \) scenarios, we can compute the expectation by summation. Then it is easy to see that we can indeed perform value evaluations on \( \hat{f} \) in polynomial time in \( s \) and \( d \).

Regarding its gradients, \( \hat{f} \) is differentiable almost-everywhere, namely for all \( x \) where \( x_i + V^\ell_i \neq ((x + V^j)_i)_{(j)} \) for all \( i, j \in [d] \) (recall that given a vector \( x \), \( x_{(i)} \) denotes its \( i \)th largest coordinate and \( x_{-i} \in \mathbb{R}^{d-1} \) is the vector \( x \) without its \( i \)th coordinate). Wherever it is differentiable, its gradient is given by

\[
\nabla_i \hat{f}(x) = \sum_{j=1}^\infty \Pr(K = j) \cdot \frac{1}{s} \sum_{\ell=1}^s 1(x_i + V^\ell_i > ((x + V^j)_{-i})_{(j)}) + \frac{2\Delta}{\delta''},
\]

and so can also be evaluated in polynomial time in \( s \) and \( d \).

**Concentration.** We will argue that \( \hat{f} \) fulfills the desirable properties if the draws \( V^1, \ldots, V^s \) sufficiently concentrate. To this end, we will use the Dvoretzky–Kiefer–Wolfowitz inequality (DKW inequality) and Chebyshev’s inequality. Both apply to distributions over the reals. We let \( X, X_1, \ldots, X_s \) be independent samples all drawn from the same distribution.

The DKW inequality [Mas90] states that the CDF and the empirical distribution are close, namely that for any \( p > 0 \) with probability at least \( 1 - p \) we have

\[
\left| \frac{1}{s} \sum_{\ell=1}^s 1(X_\ell > t) - \Pr(X > t) \right| \leq \sqrt{\frac{\ln(2/p)}{2s}}, \quad \forall t.
\]

Analogously, Chebyshev's inequality gives us a comparison of the expectation and the empirical average. It states that with probability at least \( 1 - p \)

\[
\left| \frac{1}{s} \sum_{\ell=1}^s X_\ell - \mathbb{E}[X] \right| \leq \sqrt{\frac{\text{Var}[X]}{p}}.
\]
Applying the DKW inequality on each of the components of \( V^1, \ldots, V^s \) and Chebyshev’s inequality on \( \max_i V_i^1, \ldots, \max_i V_i^s \), by a union bound, we can assume that with probability \( 1 - \rho \), we have

\[
\left| \frac{1}{s} \sum_{\ell=1}^s 1(V^\ell_r > t) - \Pr(\nu_r > t) \right| \leq \Delta \quad \text{for all } r \in [d],
\]

and

\[
\left| \frac{1}{s} \sum_{\ell=1}^s \max_r V^\ell_r - \Exp[\max_r \nu_r] \right| \leq \Delta',
\]

where \( \Delta = \sqrt{\frac{\ln(4d/\rho)}{2s}} \) and \( \Delta' = \sqrt{\frac{2\text{Var}[\max_r \nu_r]}{sp}} \leq \frac{4d}{ek'\sqrt{s}\rho} \). Note that for \( s \geq \max\{4d^2 \ln(4d/\rho), 16d^2/\rho\} \), we have \( 2\Delta d \leq 1 \) and \( k\Delta' \leq 1/\epsilon \).

Guarantees of the function \( \tilde{f} \). In the rest of the section, we denote the function \( \tilde{f}(\cdot) \) by \( \tilde{f}(\cdot) \) for ease of notation.

**Lemma A.3.** Whenever (18) is fulfilled, the function \( \tilde{f} \) defined in Equation (16) satisfies

\[
\tilde{f}(x) \geq \exp(-\delta') \cdot \|x\|_{\text{Top-k}} \quad \text{and} \quad \tilde{f}(x) \leq \left(1 + \frac{1}{\delta'} + \frac{2\Delta d}{\delta' \epsilon}\right) \cdot \|x\|_{\text{Top-k}} + \frac{H_d}{\delta' \epsilon} + \frac{k\Delta'}{\delta'}.
\]

Moreover, \( f \) is monotone, subadditive, and convex.

**Proof.** The proof mainly follows the steps in the proof Lemma 2.4.

The lower bound works exactly the same way. By the choice of the distributions for \( K \) and monotonicity, we have

\[
\tilde{f}(x) = \Exp_K \frac{1}{s} \sum_{\ell=1}^s \|x + V^\ell\|_{\text{Top-k}} + \frac{2\Delta}{\delta' \epsilon} \cdot \|x\|_1 \geq \Exp_K \|x\|_{\text{Top-k}} = \sum_{j=1}^\infty \Pr(K = j) \|x\|_{\text{Top-j}} \geq \Pr(K \geq k) \|x\|_{\text{Top-k}} = q^{k-1} \|x\|_{\text{Top-k}} = \left(1 - \frac{\delta'}{k}\right)^{k-1} \|x\|_{\text{Top-k}} \geq \exp(-\delta') \|x\|_{\text{Top-k}}.
\]

To obtain the upper bound, we again use the triangle inequality

\[
\tilde{f}(x) = \Exp_K \frac{1}{s} \sum_{\ell=1}^s \|x + V^\ell\|_{\text{Top-k}} + \frac{2\Delta}{\delta' \epsilon} \cdot \|x\|_1 \leq \Exp_K \|x\|_{\text{Top-k}} + \Exp_K \frac{1}{s} \sum_{\ell=1}^s \|V^\ell\|_{\text{Top-k}} + \frac{2\Delta}{\delta' \epsilon} \cdot \|x\|_1.
\]

To bound the first term, we use again \( \Exp_K \|x\|_{\text{Top-k}} \leq \|x\|_{\text{Top-k}} (1 + \frac{1}{\delta'}) \). For the third term, we naturally have \( \|x\|_1 \leq d \|x\|_{\text{Top-k}} \).

Finally, by (18), we can bound the second term using

\[
\Exp_K \frac{1}{s} \sum_{\ell=1}^s \|V^\ell\|_{\text{Top-k}} \leq \Exp_K K \frac{1}{s} \sum_{\ell=1}^s \max_i V_i^\ell \leq \frac{k}{\delta'} \cdot \left( \Exp_i \max \nu_i + \Delta' \right) = \frac{k}{\delta'} \cdot \left( \frac{H_d}{\lambda} + \Delta' \right) = \frac{H_d}{\delta' \epsilon} + \frac{k\Delta'}{\delta'}.
\]

The monotonicity, subadditivity, and convexity of \( f \) follows from the fact that the Top-k norms satisfy these properties and that \( f \) is formed by taking convex combinations of these norms.  

\( \square \)
Now for the gradient-stability of $\tilde{f}$.

**Lemma A.4.** Whenever (17) is fulfilled, for all $i$ and all non-negative $x, y \in \mathbb{R}_+^d$ where $\tilde{f}$ is differentiable, we have

$$\nabla_i \tilde{f}(x + y) \geq \exp(-\epsilon \|y\|_{\text{top-}k} - \delta' - \delta''') \cdot \nabla_i \tilde{f}(x).$$

**Proof.** We mainly follow the same steps as in the proof of Lemma 2.5, including the use of Claim 2.6. Now, we get

$$\nabla_i \tilde{f}(x + y) = \sum_{j=1}^{\infty} \Pr(K = j) \cdot \frac{1}{s} \sum_{\ell=1}^{s} 1\left(x_i + y_i + V_i^\ell > ((x + y + V^\ell)_{-i})(j)\right) + \frac{2\Delta}{\delta'''}$$

$$\geq \sum_{j=k}^{\infty} \Pr(K = j) \cdot \frac{1}{s} \sum_{\ell=1}^{s} 1\left(x_i + y_i + V_i^\ell > ((x + y + V^\ell)_{-i})(j)\right) + \frac{2\Delta}{\delta'''}$$

$$= \sum_{j=1}^{\infty} \Pr(K = j + k - 1) \cdot \frac{1}{s} \sum_{\ell=1}^{s} 1\left(x_i + y_i + V_i^\ell > ((x + y + V^\ell)_{-i})(j + k - 1)\right) + \frac{2\Delta}{\delta'''}$$

$$= q^{k-1} \cdot \sum_{j=1}^{\infty} \Pr(K = j) \cdot \frac{1}{s} \sum_{\ell=1}^{s} 1\left(x_i + y_i + V_i^\ell > ((x + y + V^\ell)_{-i})(j + k - 1)\right) + \frac{2\Delta}{\delta'''}$$

$$\geq q^{k-1} \cdot \sum_{j=1}^{\infty} \Pr(K = j) \cdot \frac{1}{s} \sum_{\ell=1}^{s} 1\left(V_i^\ell > ((x + V^\ell)_{-i})(j) + \|y\|_{\text{top-}k} - x_i\right) + \frac{2\Delta}{\delta'''}$$

where the second inequality uses Claim 2.6.

Now, we use the fact that (17) holds. This lets us relate the empirical probabilities to the CDF of an exponential distribution. (Recall $\nu_i$ is Exponentially distributed with parameter $\lambda$.) So, we get

$$\frac{1}{s} \sum_{\ell=1}^{s} 1\left(V_i^\ell > ((x + V^\ell)_{-i})(j) + \|y\|_{\text{top-}k} - x_i\right) + \frac{2\Delta}{\delta'''}$$

$$\geq \Pr\left(\nu_i > ((x + V^\ell)_{-i})(j) + \|y\|_{\text{top-}k} - x_i\right) + \frac{2\Delta}{\delta'''} - \Delta$$

$$\geq \exp\left(-\frac{\lambda \|y\|_{\text{top-}k}}{k}\right) \cdot \Pr\left(\nu_i > ((x + V^\ell)_{-i})(j) - x_i\right) + \frac{2\Delta}{\delta'''} - \Delta$$

$$\geq \exp\left(-\frac{\lambda \|y\|_{\text{top-}k}}{k}\right) \cdot \frac{1}{s} \sum_{\ell=1}^{s} 1\left(V_i^\ell > ((x + V^\ell)_{-i})(j) - x_i\right) + \frac{2\Delta}{\delta'''} - 2\Delta,$$

where the second inequality uses the property of the exponential distribution. This then gives (using $\lambda = \epsilon k$)

$$\nabla_i \tilde{f}(x + y) \geq \exp(-\epsilon \|y\|_{\text{top-}k}) \cdot q^{k-1} \cdot \sum_{j=1}^{\infty} \Pr(K = j) \cdot \frac{1}{s} \sum_{\ell=1}^{s} 1\left(V_i^\ell > ((x + V^\ell)_{-i})(j) - x_i\right) + \frac{2\Delta}{\delta'''} - 2\Delta$$

34
Theorem 2.7 (Norm Composition). Consider monotone (not necessarily symmetric) “inner” norms \( \| \cdot \|_1, \ldots, \| \cdot \|_\ell \) over \( \mathbb{R}^d \), and non-negative matrices \( A_i \in \mathbb{R}^{d \times n} \). Also consider a monotone “outer” norm \( \| \cdot \|_\infty \) over \( \mathbb{R}^\ell \), and assume that \( \| \cdot \|_\infty \leq \| \cdot \|_1 \). Then consider the composed norm over \( \mathbb{R}^n \):

\[
\text{norm}(x) := \| (A_1 x\|_1, \ldots, A_\ell x\|_\ell)\|_\infty.
\]

If each norm \( \| \cdot \|_i \) admits a \( \delta_i \)-gradient-stable approximation with error \( (\alpha_i, \gamma_i) \), then for every \( c > 0 \), \( \text{norm}(\cdot) \) admits a \( \delta \)-gradient-stable approximation with error \( (\alpha, \gamma) \), where:

\[
\alpha = \alpha_0 \cdot \left( \max_{i \geq 1} \alpha_i \right), \quad \delta = \delta_0 + \max_{i \geq 1} \delta_i, \quad \gamma = \left( \max_{i \geq 1} \alpha_i + c \right) \cdot \left[ \frac{\alpha_0}{c} \| (\gamma_1, \ldots, \gamma_\ell)\|_\infty \right] + \gamma_0.
\]

Moreover, given value and gradient oracles to the \( \delta_i \)-gradient-stable approximations of inner and outer norms, we can efficiently simulate value and gradient oracles to this \( \delta \)-gradient-stable approximation.

In order to prove this theorem we first need the following important upper bound on the gradient of gradient-stable approximations. It is well-known that we have \( \langle \nabla \| u \|, v \rangle \leq \| u \| \) (whenever the norm is differentiable). The next lemma states that still approximately holds for approximations of the norm.
Lemma A.5 (Gradient bound). Consider a convex and subadditive function $f : \mathbb{R}_+^d \to \mathbb{R}$ that approximates a norm $\| \cdot \|$ in the sense

$$\|x\| \leq f(x) \leq \alpha \|x\| + \beta \quad \forall x \in \mathbb{R}_+^d.$$ 

Then for all $u, v \in \mathbb{R}_+^d$ we have

$$\langle \nabla f(u), v \rangle \leq \alpha \|v\|.$$ 

Proof. Using subadditivity and then convexity, we have that for every $\eta \geq 0$

$$f(u) + f(\eta v) \geq f(u + \eta v) \geq f(u) + \langle \nabla f(u), \eta v \rangle,$$

which gives $\langle \nabla f(u), v \rangle \leq \frac{1}{\eta} f(\eta v)$. Further using the norm approximation properties of $f$ we get

$$\langle \nabla f(u), v \rangle \leq \frac{1}{\eta} \left( \alpha \|\eta v\| + \beta \right) = \frac{\alpha \|v\| + \beta}{\eta}.$$ 

Taking $\eta \to \infty$ then gives the desired bound. \qed

We are now ready to prove Theorem 2.7.

Proof of Theorem 2.7. To simplify the notation, let $\alpha^* = \max_{i \geq 1} \alpha_i$. Fix $\tilde{\epsilon} > 0$, and define $\epsilon_0 = \tilde{\epsilon} \frac{i}{\alpha^* + \epsilon}$ and also $\epsilon_i = \tilde{\epsilon} \frac{i}{\alpha^* + \epsilon}$ for $i \geq 1$.

Given the assumptions, for $i \geq 0$ let $f_i$ satisfy the requirements of a $\delta_i$-gradient-stable approximation of $\| \cdot \|_{(i)}$ (Definition 1.7) of error $(\alpha_i, \gamma_i)$ for $\epsilon = \epsilon_i$. Then define the function $h$ by replacing each term of $\text{norm}(\cdot)$ by its gradient-stable approximation:

$$h(x) := f_0\left(f_1(A_1x), \ldots, f_\ell(A_\ell x)\right).$$ 

(19)

We prove that $h$ satisfies all the desired properties.

$h$ is subadditive, convex, and monotone. Since each $f_i$ is monotone, it follows directly that $h$ is monotone as well. For subadditivity: To simplify the notation let $F(x) = (f_1(A_1 x), \ldots, f_\ell(A_\ell x))$.

By this definition $h(x) = f_0(F(x))$. Since each $f_i$ is subadditive, we have $f_i(A_i(u + v)) \leq f_i(A_i u) + f_i(A_i v)$, and hence $F(u + v) \leq F(u) + F(v)$ coordinate-wise. Further using the subadditivity and monotonicity of $f_0$,

$$h(u + v) = f_0(F(u + v)) \leq f_0(F(u) + F(v)) \leq f_0(F(u)) + f_0(F(v)) = h(u) + h(v),$$

which proves subadditivity of $h$.

Using the same argument, but replacing the subadditivity assumption by the convexity assumption, we see that $h$ is convex as well.
For the upper bound, the approximation property of \( f \) gives that \( f_i(A_iu) \leq \alpha_i\|A_iu\|_{(i)} + \frac{\gamma_i}{\epsilon_i} \), and hence by the monotonicity and approximation property of \( f_0 \) we have

\[
    h(u) \leq f_0\left( \alpha_1\|A_1v\|_{(1)} + \frac{\gamma_1}{\epsilon_1}, \ldots, \alpha_\ell\|A_\ell v\|_{(\ell)} + \frac{\gamma_\ell}{\epsilon_\ell} \right)
\]

\[
    \leq \alpha_0 \cdot \left\| \left( \alpha_1\|A_1v\|_{(1)} + \frac{\gamma_1}{\epsilon_1}, \ldots, \alpha_\ell\|A_\ell v\|_{(\ell)} + \frac{\gamma_\ell}{\epsilon_\ell} \right) \right\|_{(0)} + \frac{\gamma_0}{\epsilon_0}
\]

\[
    \leq \alpha_0 \cdot \left\| \left( \alpha_1\|A_1v\|_{(1)}, \ldots, \alpha_\ell\|A_\ell v\|_{(\ell)} \right) \right\|_{(0)} + \alpha_0 \cdot \left\| \left( \frac{\gamma_1}{\epsilon_1}, \ldots, \frac{\gamma_\ell}{\epsilon_\ell} \right) \right\|_{(0)} + \frac{\gamma_0(a^* + c)}{\epsilon}
\]

\[
    = \alpha \cdot \text{norm}(v) + \frac{\gamma}{\epsilon},
\]

where the next-to-last inequality uses the monotonicity of \( \| \cdot \|_{(0)} \), and the last inequality the definition of \( \alpha \) and \( \gamma \) from the statement of the theorem.

**Gradient-stability of \( h \).** Let \( A_i^j \) denote the \( j \)-th column of the matrix \( A_i \). By chain rule we have

\[
    \nabla_j h(x) = \sum_{i \geq 1} (\nabla_i f_0)(F(x)) \cdot \langle (\nabla f_i)(A_i x), A_i^j \rangle,
\]

where, for example, \( (\nabla_i f_0)(F(x)) \) denotes the \( i \)-th partial derivative of \( f_0 \) evaluated at the point \( F(x) \). Our goal is to lower bound the right-hand side when \( x = u + v \) for \( u, v \in \mathbb{R}^n_+ \).

First we observe that for all \( f_i \) and \( u, v \geq 0 \) we have \( f_i(u + v) - f_i(u) \leq \alpha_i\|v\|_{(i)} \): using convexity of \( f_i \) and then the gradient bound from Lemma A.5, we get

\[
    f_i(x + y) - f_i(x) \leq \langle \nabla f_i(x + y), y \rangle \leq \alpha_i\|y\|_{(i)}.
\]

Employing this on all the coordinates of \( F \) and using the monotonicity of \( \| \cdot \|_{(0)} \), we then get

\[
    \|F(u + v) - F(u)\|_{(0)} \leq \left\| \left( \alpha_1\|A_1v\|_{(1)}, \ldots, \alpha_\ell\|A_\ell v\|_{(\ell)} \right) \right\|_{(0)}
\]

\[
    \leq \left\| \left( \alpha^*\|A_1v\|_{(1)}, \ldots, \alpha^*\|A_\ell v\|_{(\ell)} \right) \right\|_{(0)} = \alpha^* \cdot \text{norm}(v).
\]

So using the gradient-stability of \( f_0 \) we have for all \( i \)

\[
    (\nabla_i f_0)(F(u + v)) \geq \exp \left( -\epsilon_0\|F(u + v) - F(u)\|_{(0)} - \delta_0 \right) \cdot \nabla f_0(F(u))
\]

\[
    \geq \exp \left( -\epsilon_0 \cdot \alpha^* \cdot \text{norm}(v) - \delta_0 \right) \cdot (\nabla_i f_0)(F(u)).
\]
Furthermore, the gradient-stability of each $f_i$ implies
\[(\nabla f_i)(A_i(u + v)) \geq \exp(-\epsilon_i \|A_i v\|_{(i)} - \delta_i)(\nabla f_i)(A_i u).\]

Because all entries in the matrices $A_i$ as well as the gradients $\nabla f_i$ are non-negative, we can apply these two bounds on (20) and get
\[
\nabla_j h(u + v) \\
\geq \sum_i \exp\left(-\epsilon_0 \cdot \alpha^* \cdot \text{norm}(v) - \delta_0\right) \cdot (\nabla_i f_0)(F(u)) \cdot \exp(-\epsilon_i \|A_i v\|_{(i)} - \delta_i) \cdot (\langle \nabla f_i(A_i u), A_i^j \rangle) \\
\geq \exp\left(-\epsilon_0 \cdot \alpha^* \cdot \text{norm}(v) - \delta_0 - \max_{i \geq 1}(\epsilon_i \|A_i v\|_{(i)} + \delta_i)\right) \cdot \sum_i (\nabla_i f_0)(F(u)) \cdot (\langle \nabla f_i(A_i u), A_i^j \rangle) \\
= \exp\left(-\epsilon_0 \cdot \alpha^* \cdot \text{norm}(v) - \delta_0 - \max_{i \geq 1}(\epsilon_i \|A_i v\|_{(i)} + \delta_i)\right) \cdot \nabla_j h(u). \tag{21}
\]

Moreover, given the assumption that $\| \cdot \|_{\infty} \leq \| \cdot \|_{(0)}$, we have
\[
\max_{i \geq 1}(\epsilon_i \|A_i v\|_{(i)} + \delta_i) \leq \max_{i \geq 1}(\epsilon_i \|A_i v\|_{(i)} + \max_{i \geq 1} \delta_i) \\
\leq \left\| \left(\epsilon_1 \|A_1 v\|_{(1)}, \ldots, \epsilon_{\ell} \|A_{\ell} v\|_{(\ell)}\right)\right\|_{(0)} + \max_{i \geq 1} \delta_i \\
\leq \frac{\epsilon \tilde{\epsilon}}{\alpha^* + c} \cdot \text{norm}(v) + \max_{i \geq 1} \delta_i.
\]

Plugging this in (21) and using the definitions $\epsilon_0 = \frac{\tilde{\epsilon}}{\alpha^* + c}$ and $\delta = \delta_0 + \max_{i \geq 1} \delta_i$ gives
\[
\nabla_j h(u + v) \geq \exp\left(-\epsilon_0 \cdot \text{norm}(v) - \delta\right) \cdot \nabla_j h(u),
\]
proving the gradient-stability of $h$.

**Computation.** Note that the function $h$ is given explicitly by (19) and its gradient $\nabla h$ is given explicitly by (20). Therefore, both any function value and any gradient can be computed efficiently given oracle access to the norm approximations being used in the composition.

Since these hold for every $\tilde{\epsilon} > 0$, we see that norm$(\cdot)$ admits a $\delta$-gradient-stable approximation with error $(\alpha, \gamma)$. This concludes the proof of Theorem 2.7. $\square$

### A.5 Explicit Representation of a Symmetric Norm in Polynomial Time

In this section we will use Ball-Optimization to get an explicit approximate representation of a symmetric norm.

**Lemma 2.11.** For any symmetric monotone norm $\| \cdot \|$ in $\mathbb{R}^d$ and every $\epsilon' > 0$, there is a set $W'$ of non-negative weights $w \in \mathbb{R}^d_+$ with $|W'| \leq \left(\frac{d}{\epsilon'}\right)^{O(1/\epsilon')}$ that can be computed using Ball-Optimization oracle access to $\| \cdot \|$ in time $\left(\frac{d}{\epsilon'}\right)^{O(1/\epsilon')}$. Since for all $x \in \mathbb{R}^d_+$ we have
\[
(1 - \epsilon')\|x\| \leq \max_{w \in W'} \sum_{i=1}^d w_i \|x\|_{T_{\alpha, i}} \leq (1 + \epsilon')\|x\|. \tag{7}
\]
Claim A.6. For any $v \in \mathbb{R}^d$, we have
\[
\max_{x \in \mathcal{X}} \langle x, v \rangle \leq \|v\| \leq (1 + \epsilon') \max_{x \in \mathcal{X}} \langle x, v \rangle.
\]

Proof. The first inequality follows since $\max_{x \in \mathcal{X}} \langle x, v \rangle = \|v\|$ and $\overline{\mathcal{X}}$ is only formed by zeroing some of the coordinates of non-negative vectors of $\mathcal{X}$.

For the second inequality, we can first assume that $v_1 \geq v_2 \geq \ldots \geq v_d$ since the norm $\| \cdot \|$ is symmetric. Now consider $x_s \in \mathcal{X}$ that achieves $\|v\| = \langle x_s, v \rangle$, where symmetry again implies that $x_s$ has non-increasing coordinates: $x_s(1) \geq \ldots \geq x_s(d)$. We will show that the vector $\overline{x}_s \in \overline{\mathcal{X}}$ corresponding to $x_s$ satisfies $\|v\| \leq (1 + \epsilon')(\overline{x}_s, v)$. First, observe that $x_s(1) \leq \|e_1\|$ since
\[
1 \geq \max_{v: \|v\| \leq 1} \langle x_s, v \rangle \geq \left\langle x_s, \frac{e_1}{\|e_1\|} \right\rangle = \frac{x_s(1)}{\|e_1\|}.
\]

Thus, all coordinates of $x_s$ are at most $\|e_1\|$ and are not zeroed in $\overline{x}_s$. Since coordinate of $x_s$ that is zeroed in $\overline{x}_s$ is at most $\epsilon' \frac{\|e_1\|}{d}$, we have
\[
\langle v, \overline{x}_s \rangle \geq \langle v, x_s \rangle - \left( v_i \sum_{i} e_i \cdot \epsilon' \frac{\|e_1\|}{d} \right) \geq \|v\| - \epsilon' \cdot v_1 \|e_1\| \geq \|v\| - \epsilon' \cdot \|v\|,
\]
where the last inequality uses $\|v\| \geq \|v_1 e_1\|$.

Next we sparsify the vectors in $\overline{\mathcal{X}}$ as in [CS19a] to obtain set $\mathcal{X}'$. Consider any vector $\overline{x} \in \overline{\mathcal{X}}$ with $\overline{x}_1 \geq \overline{x}_2 \geq \ldots \geq \overline{x}_d$. We first define vector $\tilde{x}$ corresponding to $\overline{x}$ where coordinate change $\tilde{x}_i > \tilde{x}_{i+1}$ is only possible when $i$ is a power of 2. (We will only prove the weaker result where we lose an extra factor of 2. But to make this factor smaller, we should consider powers of $1 + \delta$ (with floors)). Formally, define $\tilde{x}_i = x_i$ if $i = \min\{2^s, d\}$ for some integer $s \geq 0$, and otherwise if $s$ is the unique integer with $2^{s-1} < i < 2^s$ then $\tilde{x}_i = w_{\min\{2^s, d\}}$.

Claim A.7 (Claim 4.1 in [CS19a]). For any vector $v \in \mathbb{R}^d_+$ with non-increasing coordinates we have $\langle v, \tilde{x} \rangle \leq 2\langle v, x \rangle$.

Next, we obtain $x'$ from $\tilde{x}$ by rounding down each coordinate of $\tilde{x}$ to the nearest power of $(1 + \epsilon')$. The set $\mathcal{X}'$ is now defined by taking the union of all such vectors $x'$ and their $d!$ permutations. Combining with Claim A.6, this implies
\[
\max_{x \in \mathcal{X}'} \langle x, v \rangle \leq \|v\| \leq 2(1 + \epsilon')^2 \max_{x \in \mathcal{X}'} \langle x, v \rangle.
\]
Claim A.8 (Claim 5.3 in [CS19a]). There are at most \((2e)^{\max\{N,k\}}\) non-increasing sequences of \(k\) integers chosen from \(\{0,\ldots,N\}\).

Recall that each coordinate of \(x'\) is a power of \((1+\epsilon')\) and is in the range \([\frac{e'}{1+\epsilon'}\|e_1\|,\|e_1\|]\). Thus the number of distinct values for a coordinate of \(x'\) is \(O\left(\frac{\log(d/2\epsilon')}{\epsilon'}\right)\). Moreover, since coordinates in \(x'\) only change when they are a power of 2, we only need a non-increasing sequence of \(O(\log d)\) length to define \(x'\). So, by Claim A.8 the number of such \(x'\) is at most \((2e)^{\max\{O(\log(d/2\epsilon')),O(\log d)\}} = \text{poly}(d/\epsilon')^{1/\epsilon'}\).

Finally, we can define \(W'\). For any vector \(x' \in X'\) with \(x'_1 \geq x'_2 \geq \ldots \geq x'_d\), we define a weight vector \(w' \in \mathcal{W}\) as follows: starting with \(w'_d := x'_d\) let \(w'_i := x'_i - x'_{i+1}\) for \(i \in \{1,\ldots,d-1\}\). The main observation is that for any vector \(v\) with \(v_1 \geq v_2 \geq \ldots \geq v_d\),

\[
\langle v, x' \rangle = \sum_{i=1}^{d} w'_i \|v\|_{\text{Top-}i}.
\]

Thus, combining with (22), we get

\[
\max_{w \in \mathcal{W}'} \sum_{i=1}^{d} w'_i \|v\|_{\text{Top-}i} \leq \|v\| \leq 2(1+\epsilon')^2 \max_{w \in \mathcal{W}'} \sum_{i=1}^{d} w'_i \|v\|_{\text{Top-}i}.
\]

Moreover, the size of \(\mathcal{W}'\) is polynomially bounded by construction. It’s also easy to find \(\mathcal{W}'\) efficiently since we can test for any of these polynomially many candidate weight vectors \(w'\) whether they belong to \(\mathcal{W}'\): use the Ball-Optimization oracle to test whether \(\max_{\|v\| \leq 1} \langle x', v \rangle \leq 1\), where \(x'\) is the unique vector corresponding to \(w'\) with \(x'_1 \geq x'_2 \geq \ldots \geq x'_d\) as defined in the construction. 

\[\square\]

References

[AAP93] James Aspnes, Yossi Azar, Amos Fiat, Serge Plotkin, and Orli Waarts. On-line load balancing with applications to machine scheduling and virtual circuit routing. In
Proceedings of the Twenty-fifth Annual Symposium on Theory of Computing, STOC, pages 623–631, 1993.

[AAP97] James Aspnes, Yossi Azar, Amos Fiat, Serge A. Plotkin, and Orli Waarts. On-line routing of virtual circuits with applications to load balancing and machine scheduling. J. ACM, 44(3):486–504, 1997.

[AAG+95] B. Awerbuch, Y. Azar, E. F. Grove, Ming-Yang Kao, P. Krishnan, and J. S. Vitter. Load balancing in the lp norm. In Proceedings of 36th Annual Symposium on Foundations of Computer Science, FOCS, pages 383–391, 1995.

[ABC+16] Y. Azar, N. Buchbinder, T. H. Chan, S. Chen, I. R. Cohen, A. Gupta, Z. Huang, N. Kang, V. Nagarajan, J. Naor, and D. Panigrahi. Online algorithms for covering and packing problems with convex objectives. In Proceedings of 57th Annual Symposium on Foundations of Computer Science, FOCS, pages 148–157, Oct 2016.

[ABN08] Barry C. Arnold, N. Balakrishnan, and H. N. Nagaraja. A First Course in Order Statistics. Society for Industrial and Applied Mathematics, 2008.

[ACFS02] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. SIAM J. Comput., 32(1):48–77, 2002.
[AKS21] Sepehr Assadi, Thomas Kesselheim, and Sahil Singla. Improved truthful mechanisms for sub-additive combinatorial auctions: Breaking the logarithmic barrier. In Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA, pages 653–661, 2021.

[ALS+18] Alexandr Andoni, Chengyu Lin, Ying Sheng, Peilin Zhong, and Ruiqi Zhong. Subspace embedding and linear regression with orlicz norm. In Proceedings of the 35th International Conference on Machine Learning, volume 80, pages 224–233. PMLR, 10–15 Jul 2018.

[ALST14] Jacob Abernethy, Chansoo Lee, Abhinav Sinha, and Ambuj Tewari. Online linear optimization via smoothing. In Proceedings of The 27th Conference on Learning Theory, volume 35, pages 807–823. PMLR, 13–15 Jun 2014.

[ALT15] Jacob Abernethy, Chansoo Lee, and Ambuj Tewari. Fighting bandits with a new kind of smoothness. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 28. Curran Associates, Inc., 2015.

[ANN+17] Alexandr Andoni, Huy L. Nguyen, Aleksandar Nikolov, Ilya P. Razenshteyn, and Erik Waingarten. Approximate near neighbors for general symmetric norms. In Proceedings of the 49th Annual Symposium on Theory of Computing, STOC, pages 902–913. ACM, 2017.

[ANN+18] Alexandr Andoni, Assaf Naor, Aleksandar Nikolov, Ilya Razenshteyn, and Erik Waingarten. Hölder homeomorphisms and approximate nearest neighbors. In Proceedings of 59th Annual Symposium on Foundations of Computer Science, FOCS, pages 159–169, 2018.

[ANR95] Yossi Azar, Joseph Naor, and Raphael Rom. The competitiveness of on-line assignments. J. Algorithms, 18(2):221–237, 1995.

[BEY05] Allan Borodin and Ran El-Yaniv. Online computation and competitive analysis. Cambridge University Press, 2005.

[BGMS11] Avrim Blum, Anupam Gupta, Yishay Mansour, and Ankit Sharma. Welfare and profit maximization with production costs. In FOCS, pages 77–86, Nov 2011.

[Bha13] Rajendra Bhatia. Matrix analysis, volume 169. Springer Science & Business Media, 2013.

[BKS18] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksands Slivkins. Bandits with knapsacks. J. ACM, 65(3):13:1–13:55, 2018.

[BN09] Niv Buchbinder and Joseph Seffi Naor. The design of competitive online algorithms via a primal–dual approach. Foundations and Trends® in Theoretical Computer Science, 3(2–3), 2009.

[BR01] Ivan D. Baev and Rajmohan Rajaraman. Approximation algorithms for data placement in arbitrary networks. In Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, SODA, pages 661–670, USA, 2001.

[BSS18] Jaroslaw Byrka, Krzysztof Sornat, and Joachim Spoerhase. Constant-factor approximation for ordered k-median. In Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, pages 620–631, 2018.

[Car08] Ioannis Caragiannis. Better bounds for online load balancing on unrelated machines. In Proceedings of SODA, pages 972–981, 2008.

[CC76] R. A. Cody and E. G. Coffman, Jr. Record allocation for minimizing expected retrieval costs on drum-like storage devices. J. ACM, 23(1):103–115, January 1976.

[CK05] Chandra Chekuri and Sanjeev Khanna. A polynomial time approximation scheme for the multiple knapsack problem. SIAM Journal on Computing, 35(3):713–728, 2005.
Deeparnab Chakrabarty and Chaitanya Swamy. Interpolating between k-median and k-center: Approximation algorithms for ordered k-median. In 45th International Colloquium on Automata, Languages, and Programming, ICALP, volume 107, pages 29:1–29:14, 2018.

Deeparnab Chakrabarty and Chaitanya Swamy. Approximation algorithms for minimum norm and ordered optimization problems. In Proceedings of the 51st Annual Symposium on Theory of Computing, STOC, pages 126–137, 2019.

Deeparnab Chakrabarty and Chaitanya Swamy. Simpler and better algorithms for minimum-norm load balancing. In Proceedings of 27th Annual European Symposium on Algorithms, ESA, volume 144, pages 27:1–27:12, 2019.

Deeparnab Chakrabarty and Chaitanya Swamy. Approximation algorithms for minimum norm and ordered optimization problems. In Proceedings of the 51st Annual Symposium on Theory of Computing, STOC, pages 126–137, 2019.

Ashok K. Chandra and C. K. Wong. Worst-case analysis of a placement algorithm related to storage allocation. SIAM Journal on Computing, 4(3):249–263, 1975.

Yang Cai and Mingfei Zhao. Simple mechanisms for subadditive buyers via duality. In Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC, pages 170–183, 2017.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci., 9(3-4):211–407, 2014.

Reza Eghbali, Maryam Fazel, and Mehran Mesbahi. Worst case competitive analysis for online conic optimization. In 55th IEEE Conference on Decision and Control, CDC 2016, Las Vegas, NV, USA, December 12-14, 2016, pages 1945–1950. IEEE, 2016.

Michal Feldman, Nick Gravin, and Brendan Lucier. Combinatorial auctions via posted prices. In Proceedings of SODA, 2015.

Yoav Freund and Robert E Schapire. Adaptive game playing using multiplicative weights. Games and Economic Behavior, 29(1-2):79–103, 1999.

Uriel Feige and Jan Vondrák. Approximation algorithms for allocation problems: Improving the factor of 1 - 1/e. In Proceedings of FOCS, 2006.

Anupam Gupta, Amit Kumar, Viswanath Nagarajan, and Xiangkun Shen. Stochastic load balancing on unrelated machines. In Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pages 1274–1285, 2018.

Anupam Gupta, Ravishankar Krishnaswamy, and Kirk Pruhs. Online primal-dual for non-linear optimization with applications to speed scaling. In Approximation and Online Algorithms, pages 173–186. Springer Berlin Heidelberg, 2013.

Anupam Gupta, Ruta Mehta, and Marco Molinaro. Maximizing profit with convex costs in the random-order model. In 45th International Colloquium on Automata, Languages, and Programming, ICALP, volume 107, pages 71:1–71:14, 2018.

Teofilo F. Gonzalez. Handbook of Approximation Algorithms and Metaheuristics. Chapman & Hall/CRC, 2007.

Mehrdad Ghadiri, Richard Santiago, and Bruce Shepherd. Beyond submodular maximization via one-sided smoothness. In Proceedings of the Thirty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pages 1006–1025, USA, 2021.

R.A. Horn and C.R. Johnson. Matrix Analysis. Cambridge University Press, 2012.

Zhiyi Huang and Anthony Kim. Welfare maximization with production costs: A primal dual approach. In Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pages 59–72, 2015.
[IKKP15] Sungjin Im, Nathaniel Kell, Janardhan Kulkarni, and Debmalya Panigrahi. Tight bounds for online vector scheduling. In IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS, pages 525–544, 2015. 3

[IS20] Sharat Ibrahimpur and Chaitanya Swamy. Approximation algorithms for stochastic minimum-norm combinatorial optimization. In Proceedings of 61st Annual Symposium on Foundations of Computer Science, FOCS, pages 966–977, 2020. 1

[IS21] Sharat Ibrahimpur and Chaitanya Swamy. Minimum-norm load balancing is (almost) as easy as minimizing makespan. In Proceedings of 48th International Colloquium on Automata, Languages, and Programming, ICALP, volume 198, pages 81:1–81:20, 2021. 1, 2

[ISSS19] Nicole Immorlica, Karthik Abinav Sankararaman, Robert E. Schapire, and Aleksandrs Slivkins. Adversarial bandits with knapsacks. In Proceedings of Foundations of Computer Science, FOCS, pages 202–219, 2019. 5, 24

[KMRT15] Madhukar R. Korupolu, Adam Meyerson, Rajmohan Rajaraman, and Brian Tagiku. Coupled and k-sided placements: generalizing generalized assignment. Math. Program., 154(1-2):493–514, 2015. 4

[KS20] Thomas Kesselheim and Sahil Singla. Online learning with vector costs and bandits with knapsacks. In Proceedings of Conference on Learning Theory, COLT, pages 2286–2305, 2020. 5, 6, 24, 29

[KV05] Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. Journal of Computer and System Sciences, 71(3):291–307, 2005. 8, 10, 29

[LM98] Chi-Kwong Li and Roy Mathias. Generalizations of Ky Fan’s dominance theorem. SIAM Journal on Matrix Analysis and Applications, 19(1):99–106, 1998. 14

[LNdG15] G. Laporte, S. Nickel, and F.S. da Gama. Location Science. Springer International Publishing, 2015. 1

[LNRW19] Jerry Li, Aleksandar Nikolov, Ilya Razenshteyn, and Erik Waingarten. On mean estimation for general norms with statistical queries. In Proceedings of the Thirty-Second Conference on Learning Theory, COLT, pages 2158–2172. PMLR, 2019. 1

[LT13] J. Lindenstrauss and L. Tzafriri. Classical Banach Spaces II: Function Spaces. Ergebnisse der Mathematik und ihrer Grenzgebiete. 2. Folge. Springer Berlin Heidelberg, 2013. 29

[Mas90] P. Massart. The Tight Constant in the Dvoretzky-Kiefer-Wolfowitz Inequality. The Annals of Probability, 18(3):1269 – 1283, 1990. 32

[Mol17] Marco Molinaro. Online and random-order load balancing simultaneously. In Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pages 1638–1650. SIAM, 2017. 2, 3, 29

[Mol19] Marco Molinaro. Stochastic lp load balancing and moment problems via the l-function method. In Proceedings of the Thirtieth Annual Symposium on Discrete Algorithms, SODA, pages 343–354, 2019. 4

[Mol20] Marco Molinaro. Curvature of feasible sets in offline and online optimization. CoRR, abs/2002.03213, 2020. 29

[Mol21] Marco Molinaro. Robust algorithms for online convex problems via primal-dual. In Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA, pages 2078–2092, 2021. 4, 30

[NP06] S. Nickel and J. Puerto. Location Theory: A Unified Approach. Springer Berlin Heidelberg, 2006. 1
[Rou15] Tim Roughgarden. Intrinsic robustness of the price of anarchy. J. ACM, 62(5):32:1–32:42, 2015.

[SN20] Xiangkun Shen and Viswanath Nagarajan. Online covering with \( \ell_q \)-norm objectives and applications to network design. Math. Program., 184(1):155–182, 2020.

[ST93] David B. Shmoys and Éva Tardos. An approximation algorithm for the generalized assignment problem. Mathematical Programming, 62(1):461–474, Feb 1993.

[SWY+19] Zhao Song, Ruosong Wang, Lin F. Yang, Hongyang Zhang, and Peilin Zhong. Efficient symmetric norm regression via linear sketching. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, NeurIPS, pages 828–838, 2019.

[Tha20] Nguyen Kim Thang. Online primal-dual algorithms with configuration linear programs. In Proceedings of 31st International Symposium on Algorithms and Computation, ISAAC, pages 45:1–45:16, 2020.