A. Missing Proofs from the Main Part of Section 3

Last Part of the Proof of Lemma 4. In this part we address how to repeatedly apply the conditional expectation to obtain the desired global inequality, i.e.,

$$\mathbb{E}\left[f(S)\right] \geq \tau(1-\varepsilon)^2\mathbb{E}\left[c(S)\right] .$$

In the main text we have already argued how the algorithm induces an ordering on the elements it adds to the solution, so that they can be pictured as being added one after the other. To avoid complication in the analysis we suppose that after the algorithm stops it keeps on adding dummy elements of no cost and no value, so that in total it runs for n time steps. Consider the filtration $\{\mathcal{F}_t\}_{t=1}^n$ generated by the stochastic process associated to the algorithm, where $\mathcal{F}_t$ narrates what happens up to the point when element s_t is considered. So that $\mathcal{F}_1$ is empty and $\mathcal{F}_n$ contains all the story of the algorithm except for the last—possibly dummy—added element. From the proof in the main text we know that for each time $t \in \{1,\dots,n\}$ and any possible story $\mathcal{F}_t$ of the algorithm it holds

$$\mathbb{E}\left[f(s_t \,|\, S_t) \,|\, \mathcal{F}_t\right] \geq \tau(1-\varepsilon)^2\mathbb{E}\left[c(s_t) \,|\, \mathcal{F}_t\right] , \tag{8}$$

where recall that, for each $\{\mathcal{F}_t\}_{t=1}^n$, S_t denotes the set of all elements added to the solution before s_t. Note that this claim holds also if one considers the dummy elements after the actual termination of the algorithm.

$$\begin{aligned}
\mathbb{E}\left[f(S)\right] &= \mathbb{E}\left[\sum_{t=1}^{n} f(s_t \,|\, S_t)\right] \\
&= \sum_{t=1}^{n} \mathbb{E}\left[f(s_t \,|\, S_t)\right] && (9)\\
&= \sum_{t=1}^{n} \mathbb{E}\left[\mathbb{E}\left[f(s_t \,|\, S_t) \,|\, \mathcal{F}_t\right]\right] && (10)\\
&= \mathbb{E}\left[\sum_{t=1}^{n} \mathbb{E}\left[f(s_t \,|\, S_t) \,|\, \mathcal{F}_t\right]\right] && (11)\\
&\geq \tau(1-\varepsilon)^2\mathbb{E}\left[\sum_{t=1}^{n} \mathbb{E}\left[c(s_t) \,|\, \mathcal{F}_t\right]\right] && (12)\\
&= \tau(1-\varepsilon)^2\mathbb{E}\left[\sum_{t=1}^{n} c(s_t)\right] && (13)\\
&= \tau(1-\varepsilon)^2\mathbb{E}\left[c(S)\right] .
\end{aligned}$$

Equations (9) and (11) hold by linearity of expectation, Equations (10) and (13) by the law of total expectation and finally Equation (12) follows from monotonicity of the conditional expectation and Equation (8). □

Proof of Lemma 6. Let $\mathcal{E}$ be the event $\{c(S) \geq \frac{B}{2}\}$, then

$$(1-\varepsilon)\frac{B}{2} > \mathbb{E}\left[c(S)\right] \geq \mathbb{E}\left[c(S) \,|\, \mathcal{E}\right]\mathbb{P}\left(\mathcal{E}\right) \geq \mathbb{P}\left(\mathcal{E}\right)\frac{B}{2} .$$

Hence $\mathbb{P}\left(\mathcal{E}^C\right) > \varepsilon > 0$, so repeating the experiment $\frac{1}{\varepsilon}\log(\frac{1}{\varepsilon})$ independent times is enough to have a probability at least $1-\varepsilon$ of observing at least once $\mathcal{E}^C$. □

B. Adapting ParKnapsack to Use Binary Search

As already pointed out in the main text, the value condition in ThreshSeq may exhibit a multi-modal behaviour along a single iteration of the while loop. In order to enable binary search for k^*, we want to tweak the value condition so that if it is triggered for a certain prefix A_i, it remains activated for all A_j for $j \geq i$ in the specific while loop iteration.

So, fix an arbitrary iteration of the while loop. Call S the initial solution and A the sequence drawn by SampleSeq, $\{X_i\}_i$ the sequence of the sets of elements still fitting into the budget relative to each prefix A_i and with G_i and E_i the subsets of X_i containing the *good*, i.e., marginal density greater than τ, and *bad*, i.e., negative marginal density, elements, respectively. First, note that the cost condition is clearly unimodal: the $\{X_i\}_i$ is a decreasing sequence of sets and hence $c(X_i)$ is a non-increasing sequence of costs, while $c(X)$ stays fixed: as soon as the cost of X_i drops below $(1-\varepsilon)c(X)$ it stays there for all the prefixes longer than A_i.

For the value condition we need a bit more work; if for some j it holds that

$$\varepsilon \sum_{x\in G_j} f(x \,|\, S\cup A_j) \leq \sum_{x\in E_j} \left|f(x \,|\, S\cup A_j)\right| ,$$

it may be the case that the inequality switches direction later in the same iteration of the while loop. Notice that it can happen for one of two reasons: either elements with negative marginals are added to the solution or they are thrown away due to budget constraint. We want a modification which is robust to these corner cases. To this end, we add to the value condition the absolute contribution of two sets of items.

First, we redefine the set E_i to contain also all the bad elements considered in that while loop, regardless of the budget condition, i.e.,

$$E_i \leftarrow \{a \in X : f(a \,|\, S\cup A_i) < 0\} .$$

Moreover for each prefix A_i, we define $\mathcal{E}_i$ as the set of all the items in the prefix A_i which added negative marginal when inserted in the solution, i.e.,

$$\mathcal{E}_i = \{a_t \in A_i : f(a_t \,|\, S\cup A_{t-1}) < 0\} .$$

The new value condition then says that $\sum_{x\in G_i} \varepsilon f(x \mid S \cup A_i)$ is upper bounded by

$$\sum_{x\in E_i} |f(x \mid S \cup A_i)| + \sum_{a_j \in \mathcal{E}_i} |f(a_j \mid S \cup A_j)| .$$

Notice that now everything works out just fine: the left hand side of the condition is monotonically decreasing in i, while the right hand side is monotonically increasing, by submodularity and the fact that now $\{E_t \cup \mathcal{E}_t\}_t$ is an increasing set sequence.

Algorithm 4 THRESHBIN$(X, \tau, \varepsilon, \ell, B)$ - Variant of THRESHSEQ that utilises binary search

Input: set X of elements, threshold $\tau > 0$, precision $\varepsilon \in (0, 1)$, parameter ℓ and budget B
$S \leftarrow \emptyset$; ctr $\leftarrow 0$; flag $\leftarrow 0$
$X \leftarrow \{x \in X : f(x) \geq \tau c(x)\}$
while $X \neq \emptyset$ and ctr $< \ell$ **do**
  $[a_1, a_2, \ldots, a_d] \leftarrow$ SAMPLESEQ(S, X, B)
  $b_l \leftarrow 1, b_r \leftarrow d$
  while $b_l < b_r$ **do**
    $i = \lfloor (b_r + b_l)/2 \rfloor$
    $A_i \leftarrow \{a_1, a_2, \ldots, a_i\}$
    $X_i \leftarrow \{a \in X \setminus A_i : c(a) + c(S \cup A_i) \leq B\}$
    $G_i \leftarrow \{a \in X_i : f(a \mid S \cup A_i) \geq \tau \cdot c(a)\}$
    $E_i \leftarrow \{a \in X : f(a \mid S \cup A_i) < 0\}$
    $\mathcal{E}_i \leftarrow \{a_s \in A_i : f(a_s \mid S \cup A_s) < 0\}$
    $c_1 \leftarrow c(G_i) \leq (1-\varepsilon)c(X)$
    $c_2 \leftarrow \sum_{x\in G_i} \varepsilon f(x \mid S \cup A_i) \leq \sum_{x\in E_i} |f(x \mid S \cup A_i)| + \sum_{a_j \in \mathcal{E}_i} |f(a_j \mid S \cup A_j)|$
    if c_1 or c_2 **then**
      $b_r \leftarrow i$
    else
      $b_l \leftarrow i + 1$
    if c_2 and not c_1 **then**
      flag $\leftarrow 1$
    else
      flag $\leftarrow 0$
  $k^* = b_r$
  $S \leftarrow S \cup A_{k^*}$
  $X \leftarrow G_{k^*}$
  ctr $\leftarrow$ ctr + flag
Suppose $S = \{s_1, s_2, \ldots, s_{|S|}\}$, where the indices imply the total ordering from the proof of Lemma 4
$\bar{S} \leftarrow \emptyset$
for $t = 1, \ldots, |S|$ **do**
  if $f(s_t \mid \{s_1, \ldots, s_{t-1}\}) > 0$ **then**
    $\bar{S} \leftarrow \bar{S} \cup \{s_t\}$
return $\bar{S}$

Given the new algorithm, THRESHBIN, we need to show that it retains the right properties of THRESHSEQ and argue about its adaptive and query complexity.

Lemma 7. *Consider a run of* THRESHBIN *and denote with S and $\bar{S}$ the preliminary and final solution as in the algorithm. Then the following properties hold:*

- $c(\bar{S}) \leq c(S) \leq B$
- $f(\bar{S}) \geq f(S)$
- $\mathbb{E}[f(S)] \geq \tau(1-\varepsilon)^2 \mathbb{E}[c(S)]$
- *Call G the set of elements still fitting in the budget after S, whose marginal density with respect to S is greater than τ. Then $f(\bar{S}) \geq \varepsilon\ell \sum_{x\in G} f(x \mid S)$.*

THRESHBIN *needs* $O\left(\log n \left(\frac{\log n\kappa(X)}{\varepsilon} + \ell\right)\right)$ *adaptive rounds and* $O\left(n \log n \left(\frac{\log n\kappa(X)}{\varepsilon} + \ell\right)\right)$ *value queries.*

Proof. The proof of this Lemma is quite similar to the one for THRESHSEQ, so we just highlight the differences.

First, the new value condition is stricter than the old one, so the $\mathbb{E}[f(s_t \mid S_t) \mid \mathcal{F}_t] \geq \tau(1-\varepsilon)^2 \mathbb{E}[c(s_t) \mid \mathcal{F}_t]$ inequality holds as well, where S_t and $\mathcal{F}_t$ are as in the proof of Lemma 4. This implies that

$$\mathbb{E}[f(S)] \geq \tau(1-\varepsilon)^2 \mathbb{E}[c(S)] .$$

The bounds on adaptivity and query complexity follow easily from the binary search and the analysis of Lemma 3. Further, the first and second bullets follow directly from $\bar{S} \subseteq S$ and the fact that we only filter out from S elements with negative contribution.

Consider now the last remaining statement to prove (the analog of Lemma 5). For all real numbers a we denote with $a_+ = \max\{a, 0\}$ its positive part and with $a_- = \max\{-a, 0\}$ the negative one. Clearly $a = a_+ - a_-$.

$$\begin{aligned} f(S) &= \sum_{t=1}^{T} (f(s_t \mid S_t)_+ - f(s_t \mid S_t)_-) \\ &\leq \sum_{t=1}^{T} f(s_t \mid S_t)_+ \leq \sum_{s_t \in \bar{S}} f(s_t \mid \bar{S}_t) = f(\bar{S}) , \end{aligned}$$

where in the last inequality we used submodularity.

Similarly to the proof for THRESHSEQ, consider $t_1, \ldots, t_\ell$, $E_{(1)}, \ldots, E_{(\ell)}$, $G_{(1)}, \ldots, G_{(\ell)}$, and $\mathcal{E}_{(1)}, \ldots, \mathcal{E}_{(\ell)}$. Notice that they are all disjoint. Like before, for $s_i \in S$, S_i denotes $\{s_1, \ldots, s_{i-1}\}$, but we slightly abuse the notation and have S_{t_j} denote the set S *at the end* of the iteration of the outer

while loop where ctr is increased for the ℓth time. We have

$$
\begin{aligned}
0 &\leq f\big(S_{t_\ell} \cup \bigcup_{j=1}^{\ell} E_{(j)}\big) \leq f(S_{t_\ell}) + f\big(\bigcup_{j=1}^{\ell} E_{(j)} \,|\, S_{t_\ell}\big) \\
&\leq \sum_{s_i \in S_{t_\ell}} \left(f(s_i \,|\, S_i)_+ - f(s_i \,|\, S_i)_-\right) + \sum_{j=1}^{\ell} f(E_{(j)} \,|\, S_{t_j}) \\
&\leq \sum_{s_i \in S_{t_\ell}} \left(f(s_i \,|\, S_i)_+ - f(s_i \,|\, S_i)_-\right) \\
&\quad + \sum_{j=1}^{\ell} \sum_{x \in E_{(j)}} f(x \,|\, S_{t_j})\,.
\end{aligned}
$$

Rearranging terms, and using the value condition, we get

$$
\begin{aligned}
f(\bar{S}) &\geq \sum_{s_i \in S_{t_\ell}} f(s_i \,|\, S_i)_+ \\
&\geq \sum_{s_i \in S_{t_\ell}} f(s_i \,|\, S_i)_- + \sum_{j=1}^{\ell} \sum_{x \in E_{(j)}} |f(x \,|\, S_{t_j})| \\
&\geq \sum_{j=1}^{\ell} \left[\sum_{x \in E_{(j)}} |f(x \,|\, S_{t_j})| + \sum_{s_i \in \mathcal{E}_{(j)}} |f(s_i \,|\, S_i)| \right] \\
&\geq \varepsilon \sum_{j=1}^{\ell} \left[\sum_{x \in G_{(j)}} f(x \,|\, S_{t_j}) \right] \\
&\geq \ell\varepsilon \sum_{x \in G_{(\ell)}} f(x \,|\, S_{t_\ell})\,.
\end{aligned}
$$

Observing that $S_{t_\ell} = S$ concludes the proof. □

Theorem 5. *For $\varepsilon \in (0, 1/3)$, it is possible to achieve a $(9.465 + \varepsilon)$-approximation in $O(\frac{1}{\varepsilon}\log^2 n)$ adaptive rounds and $O(\frac{n}{\varepsilon^3}\log^3 n \log\frac{1}{\varepsilon})$ queries.*

Proof. A large part of this proof is very similar to the proof of Theorem 1. Hence, we only highlight the differences, while retaining the same notation. Note that now S is no more the output of the algorithm, but the non-filtered output of ThreshBin (the filtered version being $\bar{S}$). The two cases in the analysis are similar.

If $\mathbb{E}\left[c(S)\right] \geq (1-\varepsilon)\frac{B}{2}$, then we have a result analogous to Equation (6) in the main text:

$$
\begin{aligned}
ALG &\geq \mathbb{E}\left[f(\bar{S})\right] \geq \mathbb{E}\left[f(S)\right] \\
&\geq \tau(1-\varepsilon)^2 \mathbb{E}\left[c(S)\right] \\
&\geq \frac{1}{2}\alpha(1-\varepsilon)^4 f(O)\,.
\end{aligned}
$$

Otherwise, we can repeat the algorithm $\frac{1}{\varepsilon}\log(\frac{1}{\varepsilon})$ times to be sure that, with probability at least $1-\varepsilon$, we observe $\frac{B}{2} > c(S) > c(\bar{S})$. From this we can infer that at most one element is contained in $\tilde{G} \cap O_H$. Notice however a difference here: G and $\tilde{G}$ are the elements with good marginal with respect to S, not with respect to $\bar{S}$.

$$
\begin{aligned}
f(S \cup O_H) &\leq f(S) + \mathbf{1}_{\mathcal{E}} \cdot f(\tilde{x} \,|\, S) \\
&\quad + \sum_{x \in G} f(x \,|\, S) + \sum_{x \in O_H \setminus (G \cup \tilde{G})} f(x \,|\, S) \\
&\leq f(\bar{S})(1+\hat{\varepsilon}) + \mathbf{1}_{\mathcal{E}} \cdot \left(f(\tilde{x} \,|\, S) - \tau c(\tilde{x})\right) + \tau c(O_H) \\
&\leq f(\bar{S})(1+\hat{\varepsilon}) + \mathbf{1}_{\mathcal{E}} \cdot \left(f(x^*) - \tau\tfrac{B}{2}\right) + \tau c(O_H)\,,
\end{aligned}
$$

where $\mathcal{E}$ is the event that $\tilde{G} \cap O_H$ is not empty given that $c(S) < \frac{B}{2}$. Proceeding as in the proof of Theorem 1 and noting that $f(S) \leq f(\bar{S}) \leq ALG$ we arrive at the same inequality as in Equation (7) of the main text:

$$
f(O) \leq \frac{(1+q+\hat{\varepsilon})}{\left[p(1-p) - \alpha p + \frac{\alpha q}{2} - 2\hat{\varepsilon}\right]} ALG\,.
$$

The rest of the proof is essentially the same with the proof of Theorem 1. □

C. Monotone Objectives and a Knapsack Constraint

For monotone objectives we can improve the approximation factor by slightly modifying the main algorithm. Notice, moreover, that in ThreshSeq the only relevant condition is the cost condition since no element can have a negative marginal value.

Lemma 8. *For any set X, threshold τ, precision $\varepsilon \in (0,1)$, parameter ℓ and budget B, the random set S output by* ThreshSeq *is such that*

$$
\mathbb{E}\left[f(S)\right] \geq \tau(1-\varepsilon)\mathbb{E}\left[c(S)\right].
$$

The random set S is always a feasible solution and if $c(S) < B$, then all the elements in $X \setminus S$ have either marginal density with respect to S smaller than τ or there is no room for them in the budget. Finally, the adaptivity is upperbounded by $\frac{1}{\varepsilon}\log\left(n\kappa(X)\right)$, while the query complexity by $\frac{n^2}{\varepsilon}\log\left(n\kappa(X)\right)$.

Proof. Again the proof is similar to the one for the non-monotone case in the main text. There are three main differences. First, the adaptive complexity is given only by the number of times the cost condition is triggered, hence an upper bound is given by $\frac{1}{\varepsilon}\log\left(n\kappa(X)\right)$. The query complexity is simply obtained multiplying that by a n^2 factor as in the proof of Theorem 1.

Second, the algorithm can now only stop if the budget is exhausted or there are no good elements fitting within the budget; the while loop terminates only in those two cases.

Finally, the main chain of inequalities is simply

$$\begin{aligned}\mathbb{E}[f(s_t \mid S_t) \mid \mathcal{F}_t] &= \sum_{x \in X} p_x f(x \mid S_t) \\ &\geq \tau \sum_{x \in G} p_x c(x) \\ &\geq \tau(1-\varepsilon) \sum_{x \in X} p_x c(x) \\ &= (1-\varepsilon)\tau \mathbb{E}\left[c(s_t) \mid \mathcal{F}_t\right] ,\end{aligned}$$

where in the second inequality we used the fact that the cost condition is not triggered. Taking the expectation on the whole process and reasoning as in appendix A, we conclude the proof. □

We are ready to present the full algorithm for the monotone case. There are two main differences to the non-monotone case. First, there is no need to sample a subset H and use the Sampling Lemma, since $f(S) \leq f(S \cup O)$. Second, if one defines the small elements to be the ones with cost smaller than $\varepsilon \frac{B}{n}$ it is possible to account for them by simply adding all of them to the solution at the cost of filling an ε fraction of the budget. Notice that this can be done while keeping $\kappa(\mathcal{N}_+)$ linear in n. The remaining $(1-\varepsilon)$ fraction of the budget is then filled via THRESHSEQ on the large elements. The pseudocode is given in Algorithm 5 below.

Algorithm 5 PARKNAPSACKMONOTONE - Full algorithm for monotone and a knapsack constraint

Input: Ground set $\mathcal{N}$, monotone submodular function f, budget B, precision $\varepsilon \in (0,1)$ and par. $\alpha \in (0,1)$
$\mathcal{N}_- \leftarrow \{x \in \mathcal{N} : c(x) < \varepsilon \frac{B}{n}\}$
$\mathcal{N}_+ \leftarrow \mathcal{N} \setminus \mathcal{N}_-$
$x^* \leftarrow \max_{x \in \mathcal{N}} f(x),\ \hat{\tau} \leftarrow \alpha n \frac{f(x^*)}{B}$
$\hat{\varepsilon} \leftarrow \frac{1}{10}\varepsilon,\ k \leftarrow \frac{1}{\hat{\varepsilon}} \log(n)$
for $i = 0, \ldots, k$ in parallel **do**
  $\tau_i \leftarrow \hat{\tau} \cdot (1-\hat{\varepsilon})^i$
  for $j = 1, \ldots, \frac{1}{\hat{\varepsilon}} \log(\frac{1}{\hat{\varepsilon}})$ in parallel **do**
    $S_j^i \leftarrow$THRESHSEQ$(\mathcal{N}_+, \tau_i, \hat{\varepsilon}, (1-\hat{\varepsilon})B)$
    $T_j^i \leftarrow S_j^i \cup \mathcal{N}_-$
$T \leftarrow \arg\max\{f(T_i^j), f(x^*)\}$
Return T

Theorem 6. *For $\varepsilon \in (0,1)$ it is possible to achieve a $3+\varepsilon$ approximation in $O(\frac{1}{\varepsilon}\log n)$ adaptive rounds and $O(\frac{n^2}{\varepsilon^3}\log^2 n \log\frac{1}{\varepsilon})$ queries* or *in $O(\frac{1}{\varepsilon}\log^2 n)$ adaptive rounds and $O(\frac{n}{\varepsilon^3}\log^3 n \log\frac{1}{\varepsilon})$ queries.*

Proof. We show that PARKNAPSACKMONOTONE with parameters $\alpha = \frac{2}{3}$ and any $\varepsilon \in (0,1)$ satisfies the statement of the theorem. We start by noting that the adaptivity bound is given by combining Lemma 8, the fact that the thresholds are guessed in parallel, and the fact that $\kappa(\mathcal{N}_+) \in O(\frac{n}{\varepsilon})$. We remark that now, since the cost condition is unimodal, binary search in THRESHSEQ works without any major adjustment.

Let O^* be the optimal solution and let $\tau^* = \alpha \frac{f(O^*)}{B}$. By the parallel guesses we have that there exists a $\tau = \tau_i$ such that $(1-\hat{\varepsilon})\tau^* \leq \tau < \tau^*$. As in the non-monotone case, this is because $f(x^*) \geq f(O^*) \geq f(x^*)$. Let S be the random set outputted by THRESHSEQfor that τ. Also, let $T = S \cup \mathcal{N}_-$ and notice that $c(S \cup \mathcal{N}_-) \leq B$.

We can distinguish two cases. First, if $\mathbb{E}\left[c(S)\right] \geq \frac{B}{2}(1-2\hat{\varepsilon})(1-\hat{\varepsilon})$, then we apply Lemma 8 and we have

$$\begin{aligned}f(O^*) &\leq \frac{2}{\alpha(1-\hat{\varepsilon})^3(1-2\hat{\varepsilon})} f(S) \\ &\leq \frac{2}{\alpha(1-\hat{\varepsilon})^3(1-2\hat{\varepsilon})} f(T) . \end{aligned} \tag{14}$$

Let's now address the other case. We can argue as we did in the monotone case: if we run $\frac{1}{\hat{\varepsilon}}\log(\frac{1}{\hat{\varepsilon}})$ independent times the algorithm, at least one of them respects $c(S) < B(\frac{1}{2} - \hat{\varepsilon})$, with probability at least $(1-\hat{\varepsilon})$ (the proof of this fact is practically the same as the one of Lemma 6). Let's call $\mathcal{G}$ that event, similarly to what we have done in the main text. Focus on that run and call $\mathcal{E}$ the event that in that run there is a good element with respect to the solution not fitting in the budget. Clearly there may be at most one such item which belongs to the optimal solution O^*; we call such element $\tilde{x}$. If $\tilde{x}$ exists, then $c(\tilde{x}) \geq \frac{B}{2}$. This is because in the budget of THRESHSEQ, i.e., $B(1-\hat{\varepsilon})$, at least $\frac{B}{2}$ budget is empty, under $\mathcal{G}$.

$$\begin{aligned}f(O^*) &\leq f(S \cup O^*) = f(T \cup (O^* \setminus \mathcal{N}_-)) \\ &\leq f(T) + \mathbf{1}_{\mathcal{E}} f(\tilde{x} \mid T) + \sum_{x \in O^* \setminus \{\tilde{x}\}} f(x \mid T) \\ &\leq f(T) + \mathbf{1}_{\mathcal{E}} f(x^*) + \sum_{x \in O^* \setminus \{\tilde{x}\}} f(x \mid S) \\ &\leq f(T) + \mathbf{1}_{\mathcal{E}}(f(x^*) - \alpha\tfrac{f(O^*)}{2}) + \alpha f(O^*) .\end{aligned}$$

Passing to the expectation and calling q the conditioned probability of the event $\mathcal{E}$ given $\mathcal{G}$ we have, similarly to the main text:

$$\begin{aligned}&f(O^*) \\ &\leq \mathbb{E}\left[f(O^* \cup S)\right] \\ &= \mathbb{E}\left[f(O^* \cup S) \mid \mathcal{G}\right] \mathbb{P}\left(\mathcal{G}\right) + \mathbb{E}\left[f(O^* \cup S) \mid \mathcal{G}^C\right] \mathbb{P}\left(\mathcal{G}^C\right) \\ &\leq \mathbb{E}\left[f(O^* \cup S) \mid \mathcal{G}\right](1-\hat{\varepsilon}) + \hat{\varepsilon} 2 f(O^*) \\ &\leq (1+q)(1-\hat{\varepsilon}) ALG + (2\hat{\varepsilon} + \alpha(1-\hat{\varepsilon})(1-\tfrac{q}{2})) f(O) .\end{aligned}$$

Notice we used the bound $f(S \cup O^*) \leq 2f(O^*)$ which is universal as long as $c(S) \leq B$. Rearranging the terms we

have

$$f(O^*) \leq \frac{(1+q)(1-\hat{\varepsilon})}{1-2\hat{\varepsilon}-\alpha(1-\hat{\varepsilon})(1-\frac{q}{2})} ALG\,. \tag{15}$$

Putting together Equations (14) and (15) of this appendix and setting $\alpha = \frac{2}{3}$, we have

$$OPT \leq (3+10\hat{\varepsilon})ALG\,,$$

for any value of q. Rescaling $\hat{\varepsilon}$ by a factor of 10 one yields the desired result. □

D. Non-Monotone Objectives and a Cardinality Constraint

In presence of cardinality constraints, there is no need to address separately small and large elements. Moreover, when bounding the elements of the solution whose marginal density is greater than τ but do not fit in the budget, we just need to consider the case $\mathbb{E}[c(S)] > (1-\hat{\varepsilon})k$ instead of considering half of the "budget".

If $\mathbb{E}[|S|] \geq (1-\hat{\varepsilon})k$ we have immediately a good bound in expectation. Otherwise, if we run it at least $\frac{1}{\hat{\varepsilon}}\log(\frac{1}{\hat{\varepsilon}})$ independent times, we have that, with probability at least $(1-\hat{\varepsilon})$ the cardinality constraint k is not met, meaning that all the good elements belong to the set G, as defined in the main text. The full algorithm, ParCardinal, is presented in Algorithm 6 below.

Algorithm 6 ParCardinal - Full algorithm for non-monotone and a cardinality constraint

Input: Ground set $\mathcal{N}$,submodular function f, cardinality k, precision $\varepsilon \in (0,1)$, parameter $\alpha \in (0,1)$ and sampling probability p
$x^* \leftarrow \max_{x\in\mathcal{N}} f(x)$, $\hat{\tau} \leftarrow \alpha n \frac{f(x^*)}{k}$
$\hat{\varepsilon} \leftarrow \frac{1}{70}\varepsilon$, $\ell \leftarrow \frac{1}{\hat{\varepsilon}^2}$, $k \leftarrow \frac{1}{\hat{\varepsilon}}\log(n)$
$H \leftarrow$ sample each element in $\mathcal{N}$ independently at random with probability p
for $i = 0, \ldots, k$ in parallel **do**
  $\tau_i \leftarrow \hat{\tau}\cdot(1-\hat{\varepsilon})^i$
  for $j = 1, \ldots, \frac{1}{\hat{\varepsilon}}\log(\frac{1}{\hat{\varepsilon}})$ in parallel **do**
    $S_j^i \leftarrow$ ThreshSeq$(H, \tau_i, \hat{\varepsilon}, \ell, k)$
$T \leftarrow \arg\max\{f(S_i^j), f(x^*)\}$
Return T

Theorem 7. *For $\varepsilon \in (0, 2/5)$ it is possible to achieve a $5.83+\varepsilon$ approximation, in $O(\frac{1}{\varepsilon}\log n)$ adaptive rounds and $O(\frac{nk}{\varepsilon^3}\log n \log k \log\frac{1}{\varepsilon})$ queries,* or *in $O(\frac{1}{\varepsilon}\log n \log k)$ adaptive rounds and $O(\frac{n}{\varepsilon^3}\log n \log^2 k \log(\frac{1}{\varepsilon}))$ queries.*

Proof. ParCardinal with parameters $\alpha = 3-2\sqrt{2}$, $p = (1-\alpha)/2$ and $\varepsilon \in (0, \frac{2}{5})$ does the job. The adaptive and query complexity are as in the knapsack case. The only difference is that now each sequence drawn from SampleSeq has at most length k. For the approximation guarantees, we consider two cases, this time depending on the relative ordering of $\mathbb{E}[|S|]$ and $(1-\hat{\varepsilon})k$.

If $\mathbb{E}[|S|] \geq (1-\hat{\varepsilon})k$, then, by Lemma 4 of the main text, we have

$$f(O^*) \leq \frac{1}{\alpha(1-\hat{\varepsilon})^4}\mathbb{E}[f(S)]\,. \tag{16}$$

Otherwise, with probability at least ε, each run of the algorithm does not fill all the cardinality constraint, meaning that with $\frac{1}{\hat{\varepsilon}}\log(\frac{1}{\hat{\varepsilon}})$ independent rounds one has that $|S| < k$ in at least one round with probability at least $1-\hat{\varepsilon}$; as usual we call $\mathcal{G}$ that event. Focus on that round and recall the definition of G from Lemma 5: G contains the good elements still fitting in the cardinality constraint. We have

$$\begin{aligned} &f(S\cup O_H) \\ &\quad\leq f(S) + \sum_{x\in G} f(x\,|\,S) + \sum_{x\in O_H\setminus(S\cup G)} f(x\,|\,S) \\ &\quad\leq f(S)(1+\hat{\varepsilon}) + \tau c(O_H)\,. \end{aligned}$$

Passing to the conditional expectation with respect to $\mathcal{G}$ and using $f(S\cup O_H) \leq 2f(O^*)$ we have

$$\begin{aligned} p(1-p)f(O^*) &\leq \mathbb{E}[f(S\cup O_H)] \\ &\leq \mathbb{E}[f(S\cup O_H)\,|\,\mathcal{G}](1-\hat{\varepsilon}) + 2\hat{\varepsilon}f(O^*) \\ &\leq \mathbb{E}[f(S)](1+\hat{\varepsilon})(1-\hat{\varepsilon}) \\ &\quad + \alpha p(1-\hat{\varepsilon})f(O^*) + 2\hat{\varepsilon}f(O^*)\,. \end{aligned}$$

By rearranging the terms, we get

$$f(O^*) \leq \frac{(1+\hat{\varepsilon})(1-\hat{\varepsilon})}{p(1-p-(1-\hat{\varepsilon})\alpha - 2\frac{\hat{\varepsilon}}{p})}\mathbb{E}[f(S)]\,, \tag{17}$$

Plugging $\alpha = 3-2\sqrt{2}$ and $p = \frac{1-\alpha}{2}$ in Equations (16) and (17) of this appendix one gets

$$OPT \leq (3+2\sqrt{2}+70\hat{\varepsilon})ALG\,.$$

If we rescale $\hat{\varepsilon} = \frac{\varepsilon}{70}$, we have the desired result. □

E. Experiments

The code of all the experiments is available at Apart from Greedy, all other algorithms need some constant parameters as part of the input, in addition to the submodular instance. The choice of ε for the ENV algorithm is discussed in the main text. For SampleGreedy and ParKnapsack, we set $p = 0.9$, without further optimizing it for each problem. Although this is not the theoretical optimal, the instances in question come from real-world datasets, which typically are not arbitrarily non-monotone.

As a result the algorithm can afford to discard elements at a lower rate, as it is less likely that these will hurt the objective in the long run. Moreover, we set $\varepsilon = 1/8$ for FANTOM and $\varepsilon = 1/8, \alpha = 2 - \sqrt{3}$ for PARKNAPSACK balancing performance and running time. Each experiment on the top row was repeated 3 times, to gain an estimate of the variance. In total, the whole array of tests ran on four `t2.micro` instances on the Amazon Elastic Compute Cloud (EC2), which is part of Amazon Web Services (AWS).

Movie Recommendation. We outline how the weights w_{ij} are created, given the MovieLens dataset. Each movie i is associated with a tag vector $t^i \in [0,1]^{1128}$, which encodes how much each tag applies to it. The tags are user generated. For example, a movie like "Titanic" could have a score of 0.9 for "ships", 0.8 for "romance" and 0 for "talking animals". These tag vectors are not normalized. Given those, we compute the similarities as:

$$w_{ij} = \sqrt{\sum_{k=1}^{1128} \left(\min\{t_k^i, t_k^j\}\right)^2}. \tag{18}$$

This approach ensures that movies with tag vectors that are close appear more similar. There are various trade-offs in the selection of the similarity metric. For instance, if one defined $w_{ij} = t^i \cdot t^j$, then a movie with all tags set to 1 would appear more similar to any other movie, even when comparing a movie with itself! Going to the other extreme, if $w_{ij} = t^i \cdot t^j / (|t^i||t^j|)$ this issue would be avoided, but some information would be lost as every movie would have a normalised tag vector, even though having a high score on one tag should not impact the score on other tags. Ultimately, using (18) appears to be a reasonable compromise between the two. We stress that our experimental findings about the performance of each algorithm remain qualitatively the same for any sensible choice of w_{ij}.