We thank all the reviewers for their time in reading our paper and providing thoughtful comments.

Reviewer 1.

- Thank you for pointing out the typo. We will fix this.

Reviewer 2.

- Regarding the adversary: we may assume that an online adaptive adversary where the adversary is allowed to see the algorithm's coin flips but only after the algorithm has played its random set. This is because our analysis never makes use of the fact that the functions are fixed by the adversary ahead of time (and similarly for online dual averaging since it is a deterministic algorithm). We note that we obtain a guarantee in expectation.
- Regarding KL divergence: The KL divergence can be defined more generally for $x, y \in \mathbb{R}^n_{>0}$ as $d_{\mathrm{KL}}(x, y) = \sum_{i=1}^n x_i \ln(x_i/y_i) - x_i + y_i$. It is not possible for any coordinate y_i to be 0 as there is no point in the dual space that may get mapped to a point with a non-positive component. One can also verify that the KL projection does not cause any coordinate to become non-positive as well. We will add the details in the revised version.
- Regarding first-order regret bound: First-order regret bounds are a type of data-dependent bounds which often depend on the magnitude of the costs. There are also "second-order" regret bounds which look at the "variance" in the sequence of cost vectors.

Reviewer 3.

- Thanks for pointing out the typo in the definition of curvature. We will fix this.
- Regarding the regret bounds in [9, 10]: The previous work [9, 10] studies online continuous DR-submodular maximization. Both regret bounds in [9, 10] involve the term $GD\sqrt{T}$, where G is the upper bound of ℓ^2-norm of the gradients of objective functions and D is the ℓ^2-diameter of the feasible set. When applying their algorithm to our setting (i.e., the objective functions are the multilinear extension and the feasible set is matroid polytope), both G and D can be $\Omega(\sqrt{n})$, where n is the size of the ground set. Hence, their bounds yields $\Omega(n\sqrt{T})$, whereas our bound is $O(\sqrt{kT\log(n/k)})$. Even if we replace online gradient descent with mirror descent in their algorithm, the regret bound is still (at least) $O(k\sqrt{T\log(n/k)})$. The improvement of factor $\sqrt{k}$ in our algorithm comes from the use of the first order regret bound in OLO, which is the main contribution of our paper. We will add the detailed comparison in the revised version.
- Regarding intuition for Equation 4.1: In the offline case, the proof of double greedy considers a potential function of the form $2f(O_i) + f(X_i) + f(Y_i)$ where O_i is OPT intersected with Y_i. The change in potential at step i of double greedy can be lower bounded by
$$-\max\{p_i^+\Delta_i^-, p_i^-\Delta_i^+\} + \frac{1}{2}(p_i^+\Delta_i^+ + p_i^-\Delta_i^-).$$
In offline double greedy, one chooses $p_i = (p_i^+, p_i^-)$ so that the above expression is non-negative, i.e. so that the potential is non-decreasing. Equation 4.1 is the online equivalent of this change in potential (albeit with a sign change).
- Sample complexity and time complexity:
  - Monotone and matroid setting: we assume that the matroid is given by the rank oracle. Our algorithm (Algorithm 6) makes $O(\frac{n^4}{\varepsilon^3}\log(\frac{n^3T}{\varepsilon}))$ calls to the evaluation oracle of the objective function f_t and solves $O(\frac{n^3}{\varepsilon})$ submodular function minimization in each round t. Note that submodular function minimization is used for the Bregman projection step.
  - Nonmonotone and unconstrained setting: Our algorithm (Algorithm 2) makes $O(n)$ calls to the evaluation oracle of the objective function f_t and makes $O(n)$ overheads in each round t. Note that our USM-balance subproblem algorithm runs in constant time since the underlying convex optimization is of constant dimension.
- On softmax extension: The softmax extension can be efficiently computed for specific submodular functions arising from determinant point processes, but it is unknown how to compute it for general submodular functions faster than multilinear extension. Since our paper assumes the value oracle model, we did not use the softmax extension. Note that the the sampling for evaluating the multilinear extension does not affect the regret bound.

Reviewer 4.

- On applications: Applications of online submodular maximization include learning blog rankings, online ad display, online resource allocation. See [9, 10, 16, 29, 30] and references therein.
- On time complexity: please see the answer to Reviewer 3.