Hackathon Docs

cosmiQ

July 28, 2021

Introduction

The main problem of interest is of the form:

$$\mathcal{H} = \sum_{t_0}^{t_f} \sum_i -\mu(t,i)\hat{w}(t,i) + \sum_{t_0+1}^{t_f} \left(\sum_i \lambda(t,i)\hat{w}(t,i) - \hat{w}(t-1,i) \right)^2 + \left(\sum_i \lambda(t_0,i)\hat{w}(t_0,i) \right)^2$$
$$+ \frac{\gamma}{2} \sum_{t_0}^{t_f} \sum_{ij} \hat{w}(t,i)\Sigma_{ij}^t \hat{w}(t,j) + \rho \left(\sum_i \hat{w}(t,i) - 1 \right)^2 \tag{1}$$

To further represent it on a quantum computer we use the following:

$$\hat{w}(t,i) = \frac{1}{K} \sum_q^{N_q} d^q \hat{n}(t,i,q), \tag{2}$$

where utilize N_q physical bits of local dimension d. For quitrits $d = 3$. We also use N_t number of time samples and N number of assets (indexed by i). Typically K is used to control the precision or resolution. For normal circumstance $K = d^{N_q}$ is sufficient. Here $\mu(t,i)$ represents the profit over the incremental timer interval of assessment. The terms $\lambda(t,i)$ control the transactions and is written in this form to allowed time-dependence as well as asset class dependence. Σ_{ij}^t are the terms in the covariance matrix and γ controls the penalty for allowing volatile assets. ρ is the Lagrange multiplier useful to control the constraint.

From which we obtain the following compact set of equations:

$$
\begin{aligned}
\mathcal{H} = & \sum_{t_0}^{t_f} \sum_{i} \sum_{q} (-\mu(t,i) - 2\rho) \frac{d^q}{K} \hat{n}(t,i,q) \\
& + \sum_{t_0}^{t_f} \sum_{i<j} \sum_{q<p} 4(\frac{\gamma}{2} \Sigma^t_{ij} + \lambda(t,i)\lambda(t,j) + \rho) \frac{d^{p+q}}{K^2} \hat{n}(t,i,q)\hat{n}(t,j,p) \\
& + \sum_{t_0}^{t_f} \sum_{i<j} \sum_{q} 2(\frac{\gamma}{2} \Sigma^t_{ij} + \lambda(t,i)\lambda(t,j) + \rho) \frac{d^{2q}}{K^2} \hat{n}(t,i,q)\hat{n}(t,j,q) \\
& + \sum_{t_0}^{t_f} \sum_{i} \sum_{q<p} 2(\frac{\gamma}{2} \Sigma^t_{ij} + \lambda(t,i)^2 + \rho) \frac{d^{p+q}}{K^2} \hat{n}(t,i,q)\hat{n}(t,i,p) \\
& + \sum_{t_0}^{t_f} \sum_{i} \sum_{q} (\frac{\gamma}{2} \Sigma^t_{ii} + \lambda(t,i)^2 + \rho) \frac{d^{2q}}{K^2} \hat{n}(t,i,q)\hat{n}(t,i,q) \\
& + \sum_{t_0}^{t_f-1} \sum_{i<j} \sum_{q<p} 4(\lambda(t,i)\lambda(t,j)) \frac{d^{p+q}}{K^2} \hat{n}(t,i,q)\hat{n}(t,j,p) \\
& + \sum_{t_0}^{t_f-1} \sum_{i<j} \sum_{q} 2(\lambda(t,i)\lambda(t,j)) \frac{d^{2q}}{K^2} \hat{n}(t,i,q)\hat{n}(t,j,q) \\
& + \sum_{t_0}^{t_f-1} \sum_{i} \sum_{q<p} 2(\lambda(t,i)^2) \frac{d^{p+q}}{K^2} \hat{n}(t,i,q)\hat{n}(t,i,p) \\
& + \sum_{t_0}^{t_f-1} \sum_{i} \sum_{q} 2(\lambda(t,i)^2) \frac{d^{p+q}}{K^2} \hat{n}(t,i,q)\hat{n}(t,i,q) \\
& + \sum_{t_0}^{t_f-1} \sum_{i<j} \sum_{q<p} (-8)(\lambda(t+1,i)\lambda(t+1,j)) \frac{d^{p+q}}{K^2} \hat{n}(t+1,i,q)\hat{n}(t,j,p) \\
& + \sum_{t_0}^{t_f-1} \sum_{i<j} \sum_{q} (-4)(\lambda(t+1,i)\lambda(t+1,j)) \frac{d^{2q}}{K^2} \hat{n}(t+1,i,q)\hat{n}(t,j,q) \\
& + \sum_{t_0}^{t_f-1} \sum_{i} \sum_{q<p} (-4)(\lambda(t+1,i)^2) \frac{d^{p+q}}{K^2} \hat{n}(t+1,i,q)\hat{n}(t,i,p) \\
& + \sum_{t_0}^{t_f-1} \sum_{i} \sum_{q} (-2)(\lambda(t+1,i)^2) \frac{d^{p+q}}{K^2} \hat{n}(t+1,i,q)\hat{n}(t,i,q) \\
& + N_t
\end{aligned}
\tag{3}
$$

These equations have been encoded in the file flatnetwork.py. That file sets up our DMRG calculations, which we use to test the quality of more approximate solutions via quantum hardware or simulation. A simpler version of the equations above can be obtained if we ignore transaction costs and volatility. This serves a useful step towards testing solutions and assessing behaviour (not to mention debugging). These equations are given by:

$$\mathcal{H} = \sum_{t_0}^{t_f} \sum_{i} \sum_{q} (-\mu(t,i) - 2\rho) \frac{d^q}{K} \hat{n}(t,i,q) + \sum_{t_0}^{t_f} \sum_{i} \sum_{q} (\rho) \frac{d^{2q}}{K^2} \hat{n}(t,i,q) \hat{n}(t,i,q)$$
$$+ \sum_{t_0}^{t_f} \sum_{i} \sum_{q<p} 2(\rho) \frac{d^{p+q}}{K^2} \hat{n}(t,i,q) \hat{n}(t,i,p)$$
$$+ \sum_{t_0}^{t_f} \sum_{i<j} \sum_{q} 2(\rho) \frac{d^{2q}}{K^2} \hat{n}(t,i,q) \hat{n}(t,j,q)$$
$$+ \sum_{t_0}^{t_f} \sum_{i<j} \sum_{q<p} 4(\rho) \frac{d^{p+q}}{K^2} \hat{n}(t,i,q) \hat{n}(t,j,p) \tag{4}$$