Valuation of Commodity Storage

Jake C. Fowler

January 2020

WARNING: THIS DOCUMENT IS CURRENTLY WORK IN PROGRESS

1 Introduction

Define v_i as the decision volume of commodity injected or withdrawn from storage at time t_i. To clarify, positive value of v_i denotes injection into storage, increasing the inventory, where as a negative value denotes the volume withdrawn. The admissible values of v_i are restricted by the minimum and maximum inject/withdraw rate functions v_{min} and v_{max}.

$$v_i \in [v_{min}(t_i, V_i), v_{max}(t_i, V_i)] \tag{1}$$

Note that v_{min} and v_{max} are both functions of time and V_i, which represents the inventory in storage at time t_i. Inventory-varying injection and withdrawal rates are commonly seen in natural gas storage, where higher storage cavern pressue results from higher inventory, which results in a higher maximum withdrawal rate, and lower maximum withdrawal rate.

The inventory V_i can be defined recursively as:

$$V_i = V_{i-1} + v_{i-1} - L(t_{i-1}, V_{i-1}) \tag{2}$$

Where $L(t_{i-1}, V_{i-1})$ is the inventory loss, as a function of time and inventory, evaluated for the previous time period. An alternative representation of inventory is as a summation:

$$V_i = V_{start} + \sum_{j=0}^{i-1} (v_j - L(t_j, V_j)) \tag{3}$$

V_{start} is the inventory at the inception. This is necessary in the case where storage capacity is purchased or leased with some amount of commodity inventory in place.

V_i is itself constrained to be within V_{min} and V_{max}, the minimum and maximum inventory functions.

$$V_i \in [V_{min}(t_i), V_{max}(t_i)] \tag{4}$$

Commonly V_{min} will evaluate to zero for all time periods, but examples where non-zero minimum inventory is needed include when regulations require a minimum level of inventory is held, as is seen for natural gas storage in some European countries. Two possible reasons why V_{max} need to be functions of time are:

- Storage could be leased for consecutive time periods, but for different notional volumes.
- The terms of leased storage commonly stipulate that the storage must be empty at the time that the leased capacity ends.

Mathematical constraints.
- Set of decision times.
- Decision volume (inject/withdraw) for each time.
- Set of admissible decision volumes which adhere to constraints.

2 Cash Flows

Define $p(v_i, t_i)$ as the net present (discounted) cash flows resulting from decision volume v_i at time t_i.

$$p(v_i, t_i, V_i) = (\mu(t_i, V_i, v_i) + v_i)s_i d(c(t_i)) - \pi(t_i, V_i, v_i) \tag{5}$$

Where:

- s_i is the spot commodity price at time t_i.
- $c(t_i)$ is the commodity settlement time function which maps from the time that commodity was delivered to the time that payment is made. In European energy markets this is usually a formulaic date in the next month.
- d is the discount factor function. Encorporating current market risk-free interest rates, this function maps from a time of a cash flow to the present value of one unit of money. Any cash flow at future time t can be multiplied by $d(t)$ in order calculate the present value of this cash flow.
- $\mu(t_i, V_i, v_i)$ is the volume of commodity consumed (not added to inventory) by the storage facility, as a function of time, inventory and decision volume. In practice this term is relevant for energy storage where some quantity of the energy commodity is consumed in order to power the motors used to get the commodity into or out of storage.
- $\pi(t_i, V_i, v_i)$ is the NPV of any other costs which are generated. An example of this in practice is the cost of running motors which facilitate injection or withdrawal.

The optimal value of the storage at time t_i with inventory V_i can be written as:

$$\Omega(t_i, V_i) = \sup_{\mathbf{v}\in\Phi} \mathbb{E}^Q \left[\sum_{j=i}^{n} p(v_j, t_j, V_j) \right] \tag{6}$$

Where $\mathbb{E}^Q$ is the expectation operator under the risk-neutral measure, $\mathbf{v}$ is an adapted decision strategy consisting of $\mathcal{F}_{t_i}$ adapted v_i at each time step, and Φ is the set of all feasible decision volume vectors, given the storage constraints presented above.

From above we can see that the storage pricing problem effectively comes down to finding the optimal decision strategy. This can be solved using Dynamic Programming. First writing the recursive Bellman equation.

$$\Omega(t_i, V_i) = \max_{v_i\in[v_{min}, v_{max}]} \left\{ p(v_i, t_i) + \mathbb{E}^Q \left[\Omega(t_{i+1}, V_i + v_i - L(t_i, V_i)) \right] \right\} \tag{7}$$

Where the time and inventory dependence of v_{min} and v_{max} has been omitted to lighten notation. The intuition behind this formula is that at every time step the optimal decision volume v_i is the one which maximises the sum of current and discounted future expected cash flows conditional upon the decision. Hence, calculating the value involves recursively solving 7, start at the end date of the storage facility and using backward induction to move back in time, at each time step calculating the optimal decision as a function of the previously calculated values at the next time step.

2.1 Valuation In Practice

In order to implement the valuation calculation using 7 approximations need to be made.

2.1.1 Bang-Bang Exercise Strategy

One problematic assumption in 7 is that the set of all permissible values of v_{min} is $[v_{min}(t_i, V_i), v_{max}(t_i, V_i)]$. The practical implemenation involves evaluating the part of 7 inside the curly brackets for all permissible values, picking the value of v_i for which this evaluates to the highest value. However, assuming $v_{min}(t_i, V_i) < v_{max}(t_i, V_i)$, there are an infinite number of permissible values for v_i making this impractical to implement. The approximating solution is to assume a "Bang-Bang" exercise strategy, this being that the decision is either to inject to storage at the maximum rate, withdraw at the maximum rate, or do nothing. Expressing this mathematically:

$$v_i \in \{v_{min}(t_i, V_i), 0, v_{max}(t_i, V_i)\} \tag{8}$$

If either $v_{min}(t_i, V_i) > 0$ or $v_{max}(t_i, V_i) < 0$ then the 0 element is removed from 8 as it is no longer in the real set of permisible values.

Using a "Bang-Bang" exercise strategy is an approximation, hence the decision stategy that this results in could be suboptimal, giving a lower value for the storage than it's true value. However, several studies have shown that this assumption is a realistic one, and that even if the set of permissible values for v_i includes a larger number of elements, the optimal strategy will be very close to the "Bang-Bang" strategy, and hence the calculated storage value will be not far from the true optimal value.

2.1.2 Inventory Space Grid

During backward induction 7 will need to be solved for specific values of the inventory V_i. Clearly this cannot be done for all valid values of V_i as there are infinite number of such values. In practice, at each time step t_i a discrete grid of values within $[V_{min}(t_i), V_{max}(t_i)]$ is chosen, and 8 is solved at each of these.

The decision of the precise elements of the inventory space grid at which 7 is evaluated is somewhat arbitrary. In practice, the following points need to be kept in mind.

- It should span the entire set of admissible inventories, i.e. the minimum value at t_i should be $V_{min}(t_i)$, and the maximum value $V_{max}(t_i)$.
- A higher number of elements, should lead to a more accurate (less suboptimal) valuation, but will take longer to run. In practice some experimentation is required to choose a grid spacing sufficiently fine to give an accurate result, but without taking too long to calculate. This will be somewhat complicated by the fact that different storage configurations will behave differently.

In the Cmdty.Storage library C# core API the calculation of this grid is abstracted away to the IDoubleStateSpaceGridCalc interface allowing an extension point where the client code to provide it's own implemenation. An implementation of this is provided which calculates the grid with a fixed spacing between elements.

The calculation of an inventory grid with superior qualities is potentially an interesting area of future study. The aim is find a good trade-off between a low number of points on this grid, in order to have low computation time, whilst maintaining a good level of accuracy such that the NPV calculated is not too suboptimal.

2.1.3 Continuation Value Interpolation

Solving 7 involves the calculating of $\mathbb{E}^Q\left[\Omega(t_{i+1}, V_i + v_i - L(t_i, V_i))\right]$ the expected value of the storage at the next time step, give the specific decision volume v_i chosen. In literature this value is often referred to as the "continuation value". During the backward induction, the continuation will always have been previously calculated. However, as described in the previous section, these values will only have been calculated at specific values of V_i on the inventory space grid. It is unlikely however, that the continuation value for the inventory equal to exactly $V_i + v_i - L(t_i, V_i)$ will have been calculated. This necessitates some sort of approximation.

- Continuation value interpolation. - Inventory space reduction.

- Problem then boils down to calculating the expectation.
- Intrinsic Valuation