Appendix: Proof of correctness

First we note that the pseudo-code uses $\mu(i)$ instead of (i) to denote the mapping from sort order to storage order, to remove ambiguity. We also use object-oriented notation $\gamma(p)$ to represent the set containing p, with $\gamma(p)$.value denoting the value field of the set object. This value field is interpreted as the absolute value of each of the elements of the set, with a negative value implying zero. The value field is stored as an ancilery array in our implementation. We also need an operation that returns the "first" element of the set, which in our datastructure is just the root of the tree that represents the set.

We start by simplifying the notation to

$$f(x) = \frac{\alpha}{\rho} \sum_{j=1}^{n} h'(j) \left| x_{(j)} \right| + \frac{1}{2} ||x - w_j||_2^2,$$

where $w_j = z_{(j)} - v_{(j)}$, and $h'(j) = h(j) - h(j-1)$. We want to find $\arg\min_x f(x)$. First we take the gradient when x_j is not at the same value as another, and is non-zero:

$$\frac{\partial f}{\partial x_{(j)}} = \frac{\alpha}{\rho} h'(j) \text{sgn}(x_{(j)}) + x_{(j)} - w_j.$$

Equating to zero gives:

$$|x_{(j)}| = |w_j| - \frac{\alpha}{\rho} h'(j),$$

when $|w_j| > \frac{\alpha}{\rho} h'(j)$, with $\text{sgn}(x_{(j)}) = \text{sgn}(w_j)$. If $|w_j| \leq \frac{\alpha}{\rho} h'(j)$, then clearly the gradient can not be equated to zero except at $x_{(j)} = 0$, where the subdifferential needs to be considered. Considering the subdifferential at zero gives the requirement that

$$\frac{w_j}{\frac{\alpha}{\rho} h'(j)} \in [-1, 1]$$

which follows from $|w_j| \leq \frac{\alpha}{\rho} h'(j)$. So essentially if $|w_j| - \frac{\alpha}{\rho} h'(j) \leq 0$ we set $x_{(j)} = 0$. This is essentially the same shrinkage as performed in the proximal operator of a L_1 regularizer.

The above update is used for singleton sets in Algorithm 2. Now notice that when several $x_{(j)}$ have the same value, say a set Q of them, and none of them are zero, then equating the gradients to zero gives a set of equations, which can be solved to give

$$\forall j \in Q, \quad \text{sgn}(w_j) x_{(j)} = \frac{1}{|Q|} \sum_{i \in Q} \left(|w_i| - \frac{\alpha}{\rho} h'(i) \right).$$

This implies that the sort ordering for equal variables doesn't effect the solution, so a stable sort is not necessary. The result for the singleton case where the value is negative follows also for the grouped together variables.

It suffices to show that at termination of Algorithm 2, these equations are satisfied by the returned x^* for all j, as this implies that 0 is in the subdifferential at x^*. The assignments in Algorithm 2 clearly ensure that this is the case under the assumption that the sort ordering of $|w_j|$ is the same as the sort order of $|x_{(j)}|$ after termination of the algorithm, and likewise for the grouped variables. So we just need to show that the assignements to the $x_{(j)}$ result in that ordering. We proceed by induction.

Assume that at iteration k of the first loop in Algorithm 2, the ordering is correct for each group containing $\mu(l) < k$, i.e.

$$\forall \mu(p), \mu(q) < k : \; |w_p| > |w_p| \implies \gamma(\mu(p)).\text{value} > \gamma(\mu(q)).\text{value}.$$

Then at iteration k, a new singleton set is created for $j = \mu(k)$, with value given by $|w_j| - \frac{\alpha}{\rho} h'(j)$. If $\gamma(j)$'s value is less than the value of $\mu(k-1)$, then by induction the ordering is correct. Otherwise, the algorithm procedes by adding j to set $\mu(k-1)$, and updating the value of j's new set to the average discussed above. This will increase the value of the set, which may cause it's value to increase above that of another set. That case is handled by the recursive merging in the while loop, in the same manner. It is clear that the while loop must terminate after no more than k merges. At termination of the while loop the ordering is then correct up to k, as all changes in the set values that would cause the ordering to change instead cause set merges.