Simulation 4

Mengqi Liu

Jul 20， 2023

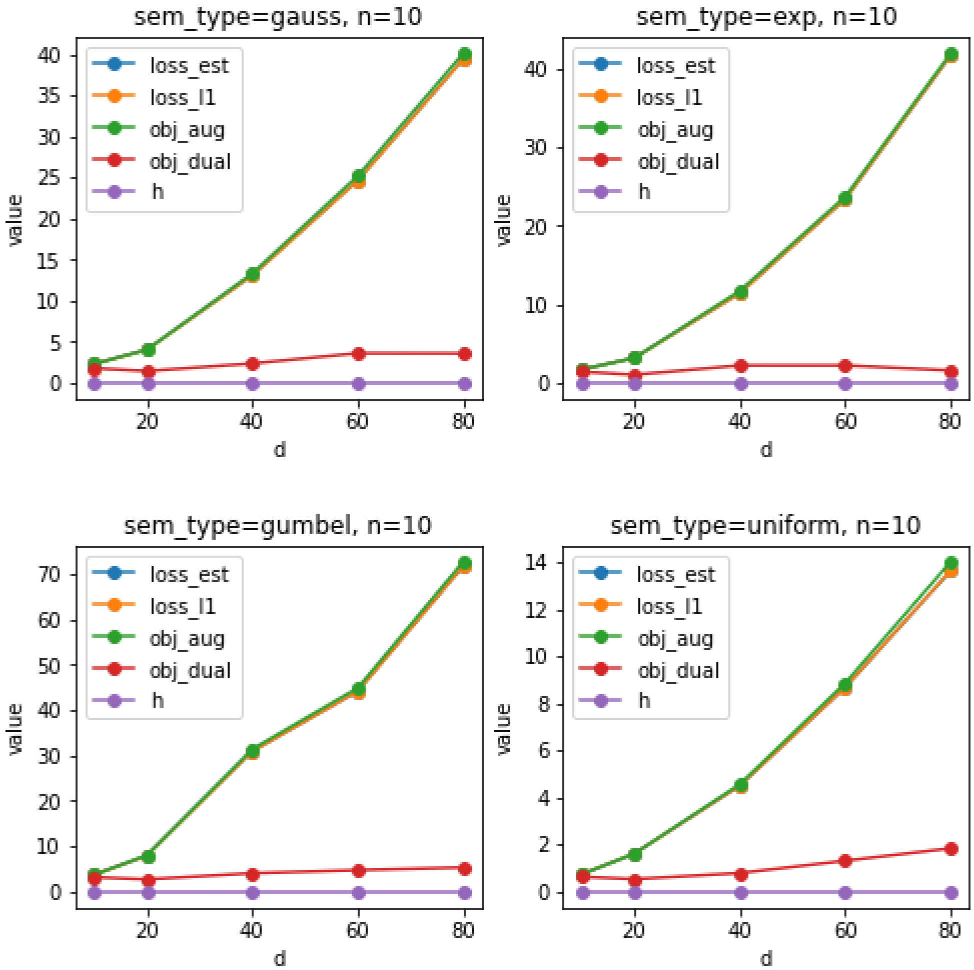
Setting

- d: number of nodes
- $s0$: expected number of edges
- graph_type: ER
  - ER: Erdős-Rényi Graph randomly choose one from all graphs with d nodes and $s0$ edges + random orientation
- n: number of samples, n=inf mimics population risk
- sem_type: gauss, exp, gumbel, uniform
- loss_type: l2
- lambda1: penalty for sparsity
- Here I use MCMC to simulate for 20 times or for 6 times.
- losses:
  - $h(W) = 0$: constraint function
  - loss_est: loss decided by loss_type
  - loss_l1: loss_est + l1 penalty
  - obj_new: + quadratic penalty of h
  - obj_dual: dual function pertaining to obj_new with Lagrange multiplier α

Results

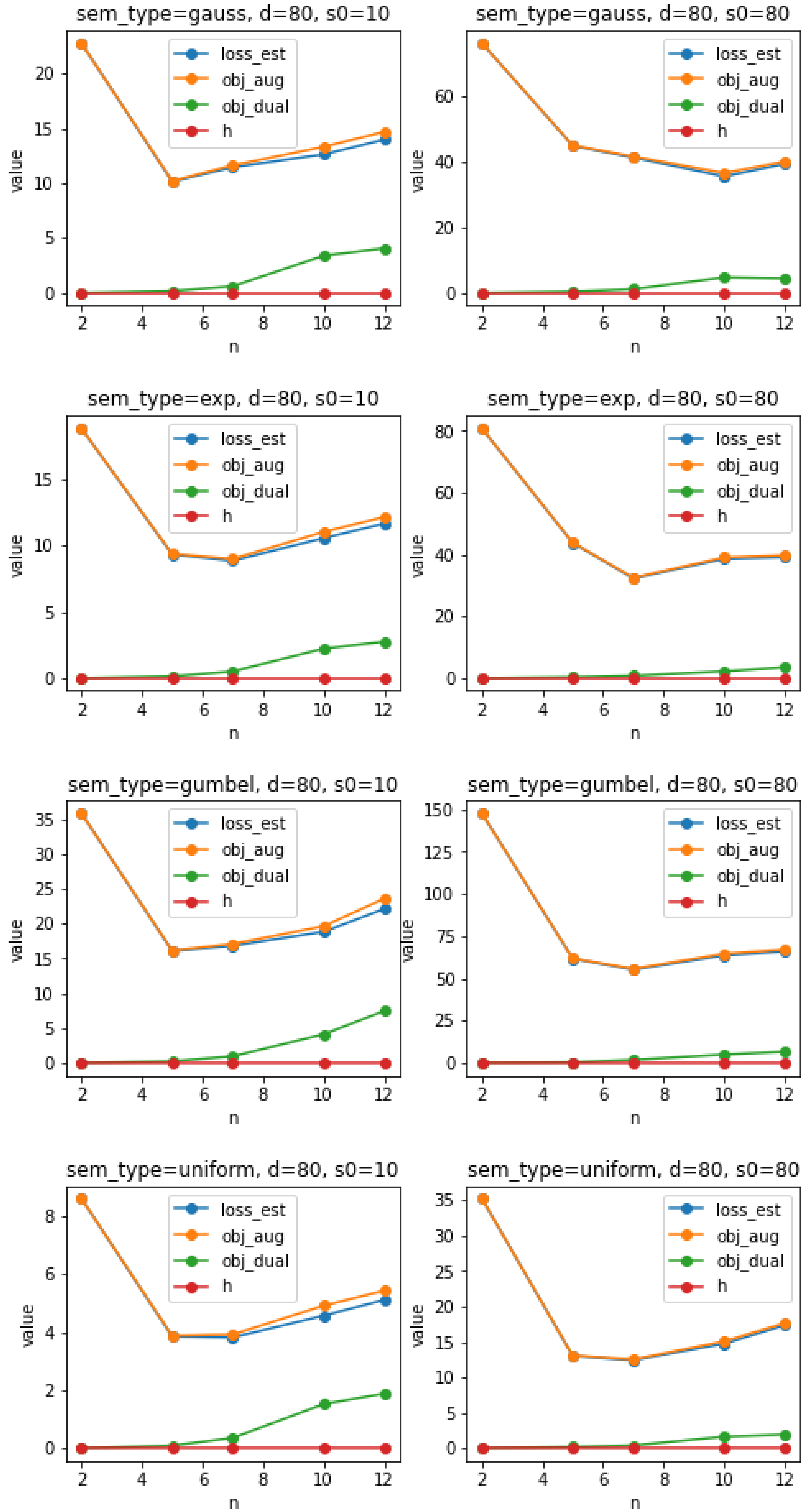
Experiment 1

Here I set lambda1 $= 0$, $d \in \{10, 20, 40, 60, 80\}$, $n = 10$ and $s0 = d$.



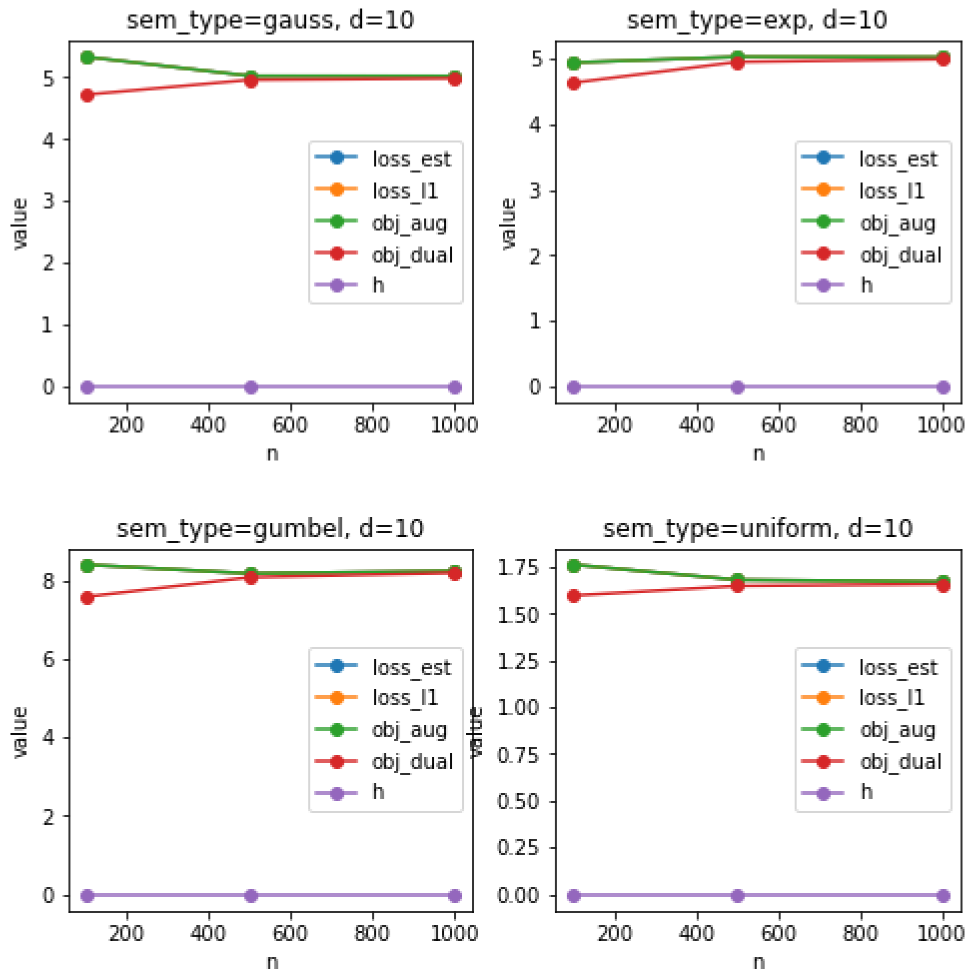
Experiment 2

Here I set lambda1 $= 0$, $n \in \{2, 5, 7, 10, 12\}$, $d = 80$ and $s0 \in \{10, 80\}$.


sem_type=gauss, d=80, s0=10
sem_type=gauss, d=80, s0=80
sem_type=exp, d=80, s0=10
sem_type=exp, d=80, s0=80
sem_type=gumbel, d=80, s0=10
sem_type=gumbel, d=80, s0=80
sem_type=uniform, d=80, s0=10
sem_type=uniform, d=80, s0=80
loss_est
obj_aug
obj_dual
h
value
n

Experiment 3

Here I set lambda1 $= 0$, $n \in \{100, 500, 1000\}$, $d = 10$ and $s0 = d$.



Gains and reflection

- When n is fixed, as the d gets bigger, the loss_est gets bigger, where obj_dual remains basically the same. ($n \leq d$)
- When d is fixed, as the n gets bigger, the loss_est gets smaller fisrt and then go bigger, where obj_dual increases very slowly. ($n \leq d$)
- When d is fixed, as the n gets extremely bigger, the loss_est and obj_dual keep constant and the difference loss_est-obj_dual seems to become smaller than in second experiment. ($n >> d$)

Question: Why doesn't loss_est not approach zero?

Technically,

- W_est[np.abs(W_est) < w_threshold] = 0 -> arbitrary threshold -> how about delete this code line? -> "2"

- scale_vec = np.zeros(d) -> this noise level may be too large to introduce more messy data -> scale_vec = np.zeros(d) * noise_scale

- if loss_type == 'l2': X = X - np.mean(X, axis=0, keepdims=True)

Theoretically,

- Considering the large gap between loss_est/obj_new and obj_dual, it may be that the systematic error of optimizing this problem is relatively large, rather than specific optimization steps.

Ideas for the next experiment:

- Explore whether arbitrary threshold for W_est will effect the losses.

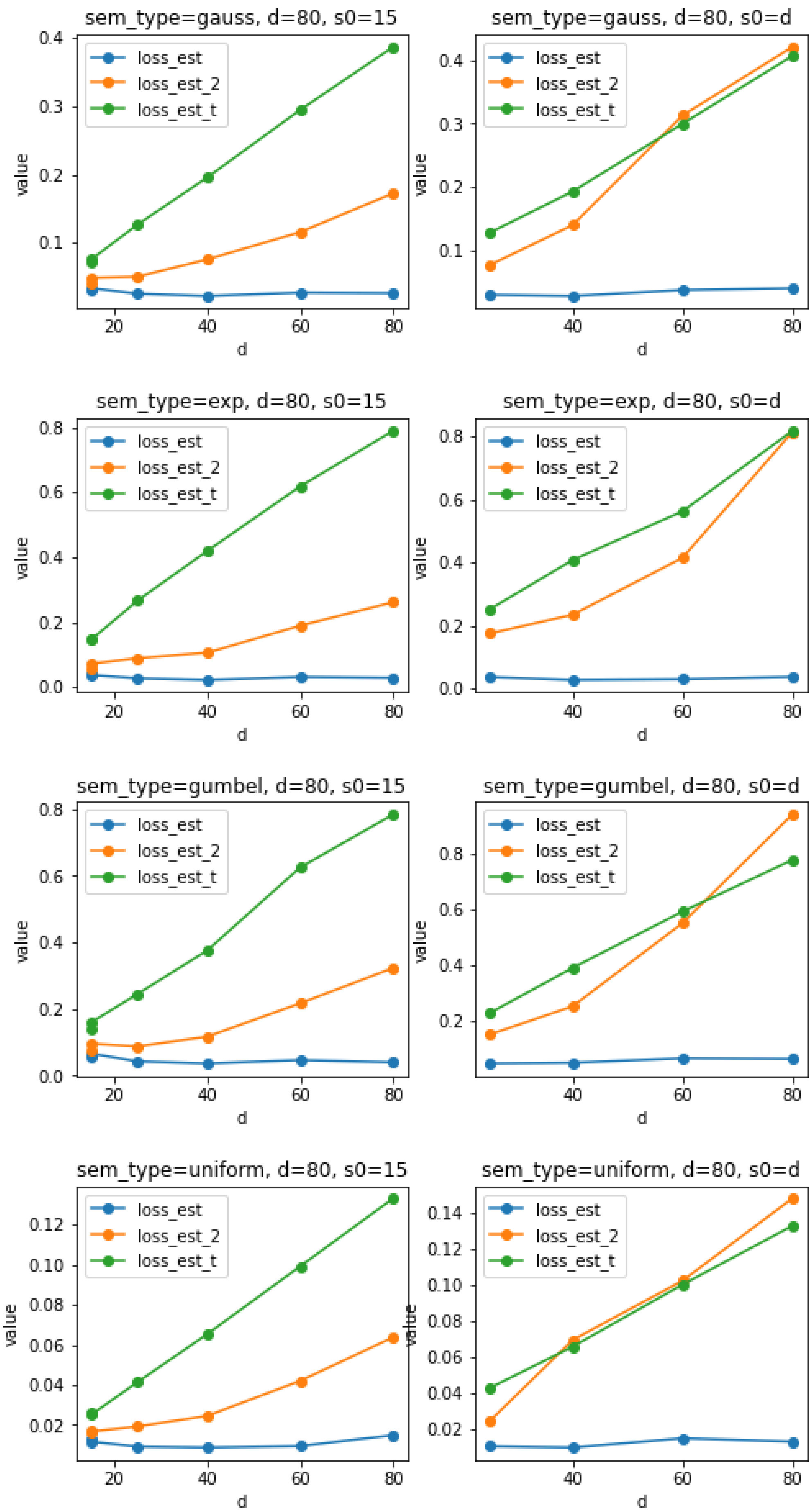
- Will losses be zero if we set noise_scale to be smaller?

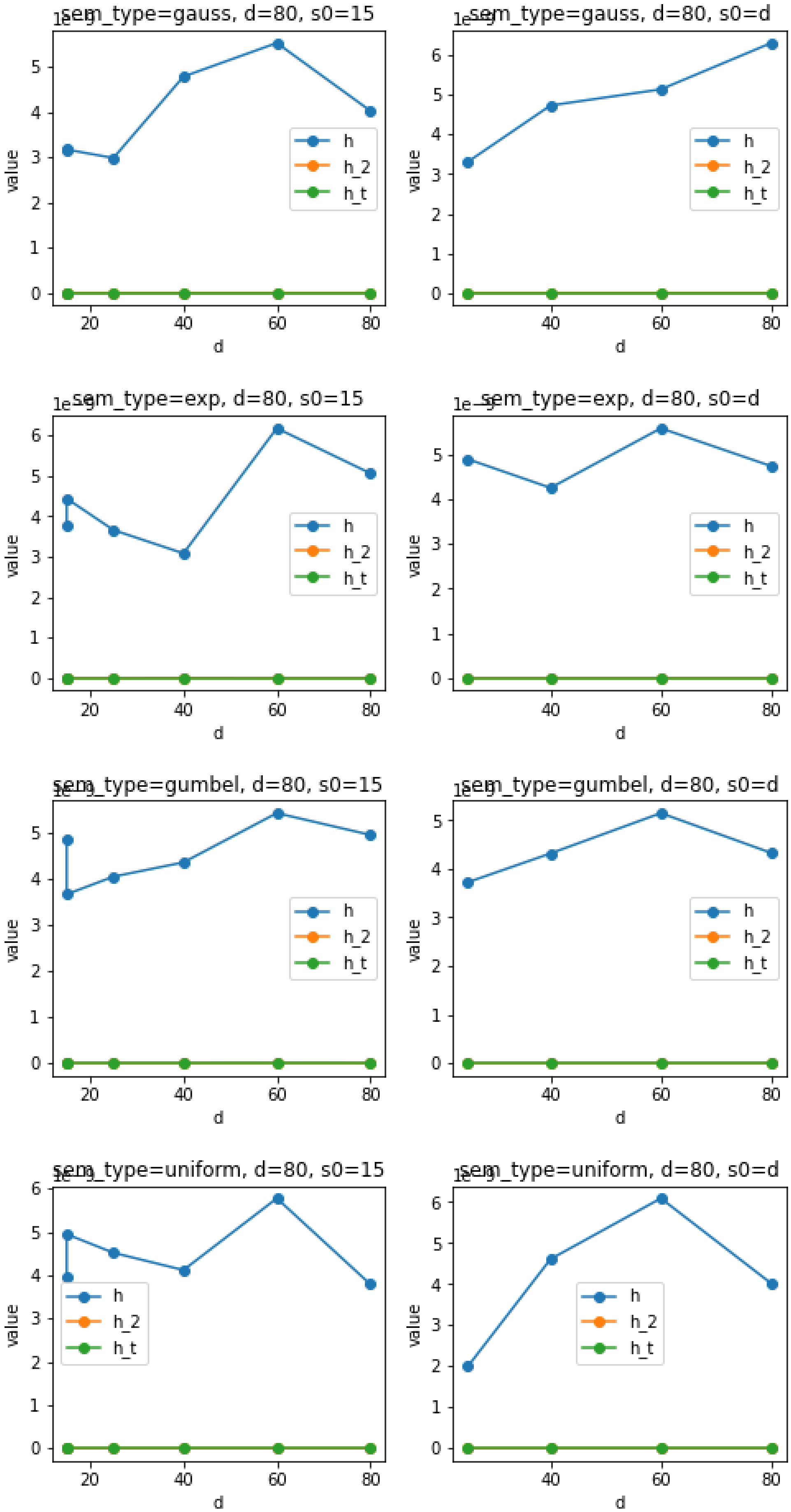
- If losses cannot achieve zero, how do they compared to losses with regard to the global true W?

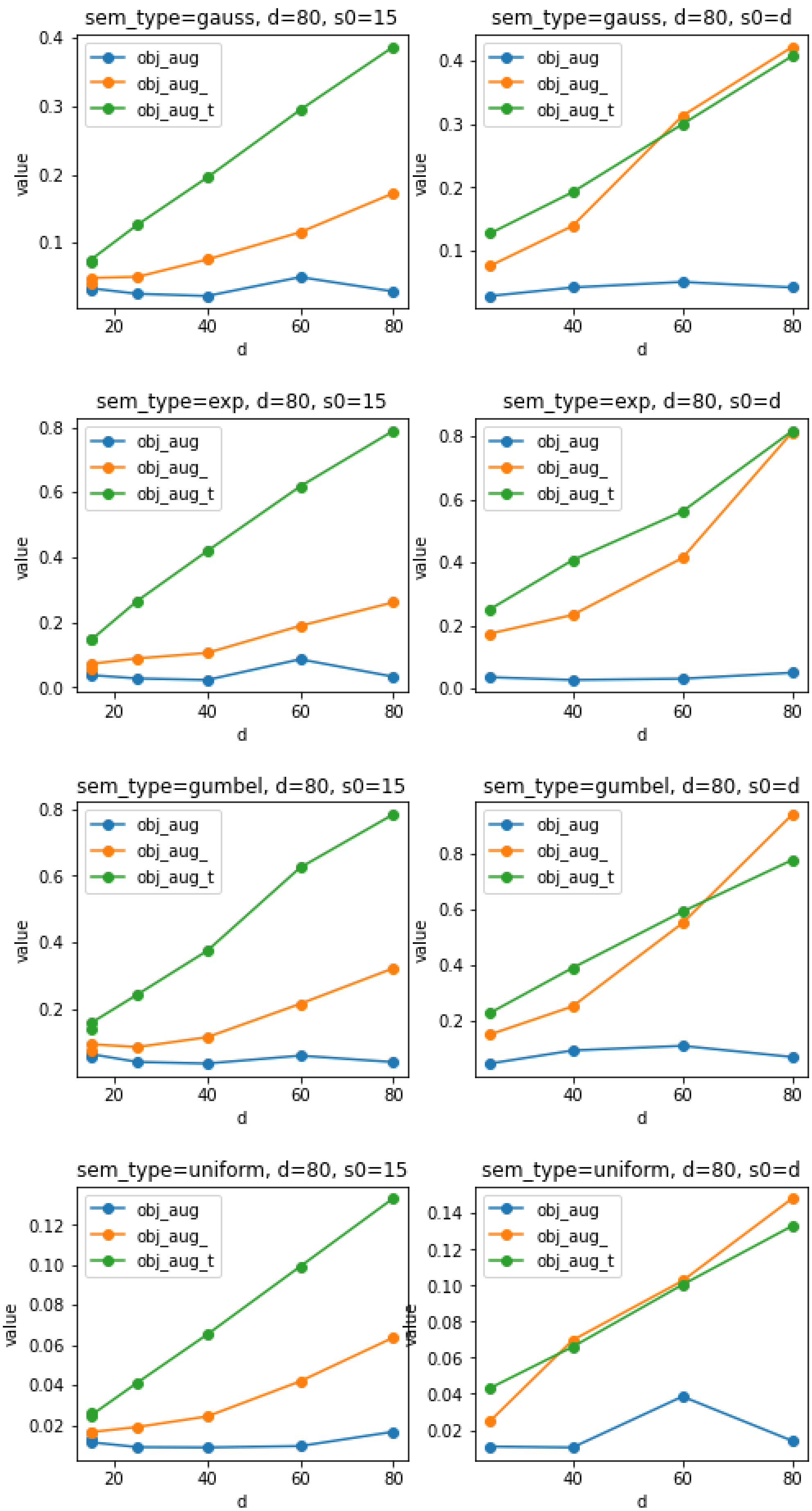
Experiment 4

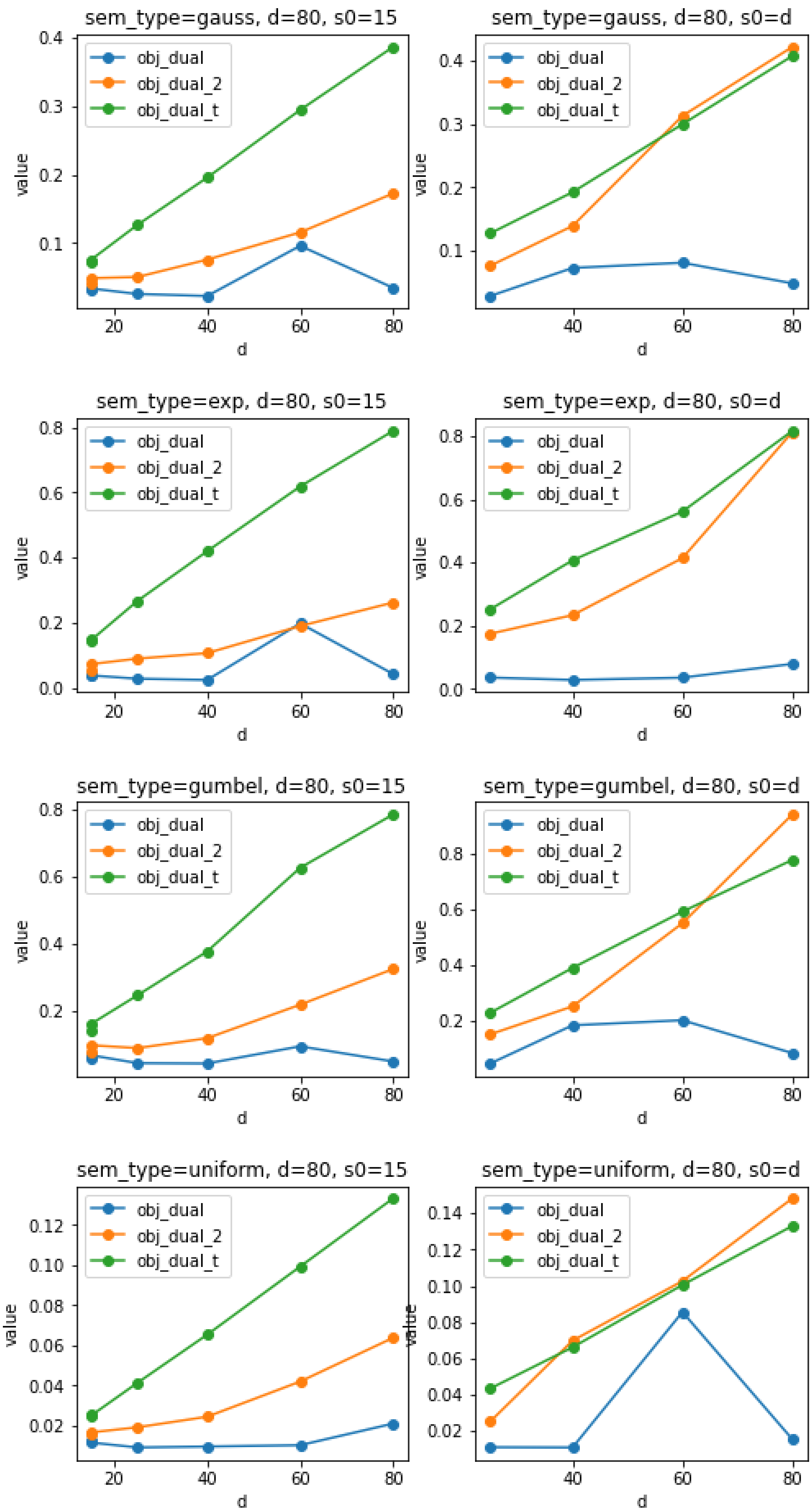
Here I set lambda1 $= 0$, $d \in \{15, 25, 40, 60, 80\}$, $n = 15$ and $s0 = \{15, d\}$.

- Blue lines stand for original W_est and its losses.
- Orange lines stand for W_est2 with "W_est[np.abs(W_est) < w_threshold] = 0" and its losses.
- Green lines stand for true W and its losses.


sem_type=gauss, d=80, s0=15
sem_type=gauss, d=80, s0=d
sem_type=exp, d=80, s0=15
sem_type=exp, d=80, s0=d
sem_type=gumbel, d=80, s0=15
sem_type=gumbel, d=80, s0=d
sem_type=uniform, d=80, s0=15
sem_type=uniform, d=80, s0=d
loss_est
loss_est_2
loss_est_t
value
d


sem_type=gauss, d=80, s0=15
sem_type=gauss, d=80, s0=d
sem_type=exp, d=80, s0=15
sem_type=exp, d=80, s0=d
sem_type=gumbel, d=80, s0=15
sem_type=gumbel, d=80, s0=d
sem_type=uniform, d=80, s0=15
sem_type=uniform, d=80, s0=d
value
d
h
h_2
h_t


sem_type=gauss, d=80, s0=15
sem_type=gauss, d=80, s0=d
sem_type=exp, d=80, s0=15
sem_type=exp, d=80, s0=d
sem_type=gumbel, d=80, s0=15
sem_type=gumbel, d=80, s0=d
sem_type=uniform, d=80, s0=15
sem_type=uniform, d=80, s0=d
obj_aug
obj_aug_
obj_aug_t
value
d


sem_type=gauss, d=80, s0=15
sem_type=gauss, d=80, s0=d
sem_type=exp, d=80, s0=15
sem_type=exp, d=80, s0=d
sem_type=gumbel, d=80, s0=15
sem_type=gumbel, d=80, s0=d
sem_type=uniform, d=80, s0=15
sem_type=uniform, d=80, s0=d
obj_dual
obj_dual_2
obj_dual_t
value
d

New gains

- Surprisingly, green lines and orange lines are more close, which means sparsely estimated W_est may behave more like true W.
- Unfortunately, losses associated with estimated matrix are smaller. This may indicate some systematic problem in optimization problem or overfitting.
- The constraint function computed by originally estimated W_est and W_est2 is not strictly 0 in some cases. The severity of the problem is uncertain.
- Smaller noise_scale will lead to smaller losses and shorter running time.