1 Log transformation

I use the notation in paper "Exact Selective Inference with Randomization" to show the calculation tricks that are potentially used in the Python code.

As in the paper, let $\phi(x)$ be the density if a standard normal variable. And $\hat{\beta}_{E,j}$ be the point estimate condition on the selection events.

$$\begin{aligned}
&log\left[\phi\left(\frac{1}{\sigma^j}(x-\lambda^j\beta_j^\varepsilon-\zeta^j)\right) * TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)\right] \\
&= log\{\frac{1}{\sqrt{2\pi}}exp(-\frac{\left(\frac{1}{\sigma^j}(x-\lambda^j\beta_j^\varepsilon-\zeta^j)\right)^2}{2})\} + log\{TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)\} \\
&= log\{\frac{1}{\sqrt{2\pi}}exp(-\frac{\left(\frac{1}{\sigma^j}(x-\lambda^j\beta_j^\varepsilon-\zeta^j+\hat{\beta}_{E,j}-\hat{\beta}_{E,j})\right)^2}{2})\} + log\{TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)\} \\
&= log\{\frac{1}{\sqrt{2\pi}}exp(-\frac{\left([x-\hat{\beta}_{E,j}]-[\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j}]\right)^2}{2{\sigma^j}^2})\} + log\{TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)\} \\
&\propto -\frac{1}{2{\sigma^j}^2}\{(x-\hat{\beta}_{E,j})^2 - 2(x-\hat{\beta}_{E,j})(\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j}) + (\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j})^2\} + log\{TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)\} \\
&= -\frac{(x-\hat{\beta}_{E,j})^2}{2{\sigma^j}^2} + \frac{(x-\hat{\beta}_{E,j})(\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j})}{{\sigma^j}^2} - \frac{(\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j})^2}{2{\sigma^j}^2} + log\{TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)\} \\
&= x\frac{\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j}}{{\sigma^j}^2} - \frac{\hat{\beta}_{E,j}(\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j})}{{\sigma^j}^2} - \frac{(\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j})^2}{2{\sigma^j}^2} + log\{TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)\} - \frac{(x-\hat{\beta}_{E,j})^2}{2{\sigma^j}^2} \\
&= x\theta + \Lambda(\theta) + log\{TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)\} - \frac{(x-\hat{\beta}_{E,j})^2}{2{\sigma^j}^2}
\end{aligned}$$

where $\theta = \frac{\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j}}{{\sigma^j}^2}$ and $\Lambda(\theta) = -\frac{\hat{\beta}_{E,j}(\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j})}{{\sigma^j}^2} - \frac{(\lambda^j\beta_j^\varepsilon+\zeta^j-\hat{\beta}_{E,j})^2}{2{\sigma^j}^2}$. Denote $logW(x) = log\{TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)\} - \frac{(x-\hat{\beta}_{E,j})^2}{2{\sigma^j}^2}$

Therefore, after obtaining the upper bound the lower bound of θ from the *equal_tailed_interval*, we need to use below expression to convert back to β_j^ε scale.

$$\beta_j^\varepsilon = \frac{\theta}{\lambda^j}{\sigma^j}^2 + \frac{\hat{\beta}_{E,j}-\zeta^j}{\lambda^j}$$

Above can be observed in the *code*.

1.1 Construct pdf

The above is log transformed expression, so when construct pdf at x_j we need take exp.

$$f(x_j;\theta) = \frac{exp(logW(x_j))exp\{x_j\theta\}exp\{\Lambda(\theta)\}}{\sum_k(exp(logW(x_k))exp\{x_k\theta\}exp\{\Lambda(\theta)\})} = \frac{exp(logW(x_j)+x_j\theta)}{\sum_k(exp(logW(x_k))exp\{x_k\theta\})} = \frac{w_j exp(x_j\theta)}{\sum_k w_k exp(x_k\theta)}$$

where $w_j = logW(x_j)$

1.2 cdf

Then the cdf is

$$P_\theta = \sum_j e^{\theta X_j - \kappa(\theta)} w_j \delta_{X_j}$$

where $\kappa(\theta) = \log\left(\sum_k w_k e^{\theta X_k}\right)$ with $\delta_{X_j} = \begin{cases} 1 & \text{if } x = x_j \\ 0 & \text{otherwise} \end{cases}$

This expression mathces the *comments* in the code.

2 Pseudocode

Below calculation process is repeated for each selected variables

2.1 pdf & cdf

- Step 1. Generate a 1000 grids for the selected variable. $[\hat{\beta}_{E,j}-6*sd, \hat{\beta}_{E,j}+6*sd]$ where sd = np.sqrt(np.diag(inverse_info)) code
- Step 2. For each grid values do:
  1. calculate $TP^{[I^j_-, I^j_+]}(\theta^j(x), \vartheta^j)$
  2. calculate $logW(x)$
- Step 3. Stabilize calculation $logW -= logW.max()$. Subtract the maximum value in logW from every element in logW.
- Step 4. Construct pdf function
  1. $M = max_{k=1,...1000}\{logW(x_k) + x_k\theta\} - 5$. *code line* 103
  2. $f(x_j;\theta) = \frac{exp(logW(x_j)+x_j\theta-M)}{\sum_k exp(logW(x_k)+x_k\theta-M)}$
- Step 5. Construct $cdf(\hat{\beta}_{E,j};\theta) = \sum_{k=1,...,1000} f(x_k;\theta)I(x_k \leq \hat{\beta}_{E,j})$

2.2 Confidence Interval

With the above pdf and cdf, now construct confidence interval

- Step 1. decide naive upper and lower bound range for θ
  1. estimate mean when $\theta = 0$, $\mu = \sum x_k f(x_k;\theta = 0)$. *Line* 516, E function
  2. estimate varaince when $\theta = 0$, $\sigma = \sqrt{\sum(x_k-\mu)^2 f(x_k;\theta = 0)}$ *line* 517, Var function
  3. naive range $[\mu - 20\sigma, \mu + 20\sigma]$
- Step 2. Further adjust the naive range to make sure target value (0.05 or 0.95) are between $cdf(\hat{\beta}_{E,j};\theta = lb)$ and $cdf(\hat{\beta}_{E,j};\theta = ub)$. code 293 - 299
- Step 3. Half split the adjusted range to find θ value
- Step 4. Convert the upper and lower bound of θ to β^{ε}_j scale. code 204 - 210

$$\beta^{\varepsilon}_j = \frac{\theta}{\lambda^j}{\sigma^j}^2 + \frac{\hat{\beta}_{E,j} - \zeta^j}{\lambda^j}$$

2.3 P value

- Step 1. calculate $\theta_0 = \frac{\lambda^j*0+\zeta^j-\hat{\beta}_{E,j}}{{\sigma^j}^2}$. code
- Step 2. $pvalue = 2 * min(cdf(\theta = \theta_0), 1 - cdf(\theta = \theta_0))$

3 Questions

1. inverse_info is the observed_info_mean in the function *solve_estimating_eqn*. But I don't know inverse_info represents which value in the paper. And what are $U1, U2, U3, U4, U5$ (they come from function *target_query_Interactspec*)?

The code has three layers: query_spec, target_spec, and target_query_interact. Query_spec is like our pre-selection period. You investigate data and let data decide which model to use. Target_spec is based on the results from data inquiry to do inference. It is like our post selection period. target_query_interact connects previous two stages. The $U1, U2, U3, U4, U5$ may not be that important in terms of replicate Python results. What important is to match Python with R in matrix mentioned in the pivot formula.

2. when calculate p value, I see you call function pivot, then _construct_density() and _construct_families() are called. And what are *prec_target*, _T, _P, *bias_target* in the construct density function? I think *prec_target_nosel* is $\frac{1}{\sigma^{j^2}}$, $prec\_target = \frac{1}{\sigma^2 |c^j|_2^2}$, $\_r = \zeta^j$, $\_S = \lambda^j$, $U_2 = {\vartheta^j}^2$, $U_3 = (P^j)^T \Omega^{-1} P^j$ in the paper.

3. Just to double check we call *selected_targets* function to obtain the *target_spec*. In the *selected_targets* function, the *observed_target* value is $\hat{\beta}_E = (X_E^T X_E)^{-1} X_E y$ the point estimate condition on selection event. and cov_target * dispersion $= \sigma^2 (X_E^T X_E)^{-1}$

4. I just notice that the matrix (like P, Q, R, $\theta^j(x), \vartheta^j$) used to construct pivot in the paper "Exact Selective Inference with Randomization" does not depend on β_j^ε. This is expected given the paper consider simple linear regression. But when consider the longitudinal data, if we take the paper "Selective Inference for Time-Varying Effect Moderation" as example, on page 15, we can observe the H and K matrix will depend on β_n^E or β_j^ε. So when construct confidence interval should I recalculate H and K matrix when I change the value for β_n^E or β_j^ε?

No. use data dependent point estimate is good enough. No need to recalculate all matrix every time when the theoretical value is updated.

5. Yiling has pointed me to the proof for monotonicity of the cumulative distribution function of a truncated Gaussian random variable. I agree the proof. My only small question is for our pivot the pdf is

$$\frac{\phi\left(\frac{1}{\sigma^j}(x - \lambda^j \beta_j^\varepsilon - \zeta^j)\right) * TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)}{\int_{-\infty}^{\infty} \phi\left(\frac{1}{\sigma^j}(x - \lambda^j \beta_j^\varepsilon - \zeta^j)\right) * TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j) dx}$$

We can write it in the exponential family form. But I am not sure whether this also follows the normal distribution given $TP^{[I_-^j, I_+^j]}(\theta^j(x), \vartheta^j)$'s mean, $\theta^j(x)$, depends on the value of x.

The theory works as long as we have a natural exponential family in the parameter of interest. In the Proof, truncated Gaussian distribution is mentioned because it is a natural exponential family so it has the monotone likelihood ratio inequality that is the core part of the proof.

Try to match with paper notation Denote ${\sigma^j}^2 = \frac{\sigma_1^2}{n + \sigma_1^2 A^T \Theta_{22}^{-1} A}$

$$\begin{aligned}
\phi(b; \beta_{\epsilon,j}, \frac{\sigma_1^2}{n}) \phi(\mathcal{Z}_{-\epsilon}; \Delta_2(b, b^{'}), \Theta_{22}) &\propto exp(-\frac{1}{2}\frac{(b - \beta_{\epsilon,j})^2}{\frac{\sigma_1^2}{n}}) exp(-\frac{1}{2}(\mathcal{Z}_{-\epsilon} - C - Ab)^T \Theta_{22}^{-1} (\mathcal{Z}_{-\epsilon} - C - Ab)) \\
&\propto exp(-\frac{1}{2\frac{\sigma_1^2}{n}}(b^2 - 2\beta_{\epsilon,j} b)) exp(-\frac{1}{2}(A^T \Theta_{22}^{-1} A b^2 - 2(\mathcal{Z}_{-\epsilon} - C)^T \Theta_{22}^{-1} A b)) \\
&= exp(-\frac{1}{2\frac{\sigma_1^2}{n}}\{(1 + \frac{\sigma_1^2}{n} A^T \Theta_{22}^{-1} A) b^2 - 2(\beta_{\epsilon,j} + \frac{\sigma_1^2}{n}(\mathcal{Z}_{-\epsilon} - C)^T \Theta_{22}^{-1} A) b\}) \\
&= exp(-\frac{1}{2}\frac{b^2 - 2\frac{\beta_{\epsilon,j} + \frac{\sigma_1^2}{n}(\mathcal{Z}_{-\epsilon} - C)^T \Theta_{22}^{-1} A}{1 + \frac{\sigma_1^2}{n} A^T \Theta_{22}^{-1} A} b}{\frac{\sigma_1^2}{n + \sigma_1^2 A^T \Theta_{22}^{-1} A}}) \\
&= exp(-\frac{1}{2{\sigma^j}^2}(b^2 - 2\{n\frac{{\sigma^j}^2}{\sigma_1^2}\beta_{\epsilon,j} + {\sigma^j}^2 (\mathcal{Z}_{-\epsilon} - C)^T \Theta_{22}^{-1} A\} b))
\end{aligned}$$

$\lambda^j = n\frac{{\sigma^j}^2}{\sigma_1^2}$ and $\zeta^j = {\sigma^j}^2 (\mathcal{Z}_{-\epsilon} - C)^T \Theta_{22}^{-1} A$