Extending the use of statistical emulators in Bayesian experimental design

James McGree
Associate Professor of Statistics
Queensland University of Technology
james.mcgree@qut.edu.au, jamesmcgree.com, @j_mcgree

Joint work with
Antony Overstall
Associate Professor of Statistics
University of Southampton, UK.

April 18, 2018

Introduction

Aim

- Efficiently derive high dimensional Bayesian designs
- Large scale model and parameter uncertainty

Methodology

- Extend the use of emulators in Bayesian design
- Propose the new k-dimensional approximate coordinate exchange algorithm

Motivation

- Current methods can be inefficient
- Extensions are needed in general but particularly for high dimensional designs such as screening experiments

Bayesian inference

- Predominately focused the posterior distribution:

$$p(\boldsymbol{\theta}_m|\boldsymbol{y}, \boldsymbol{d}, m) = p(\boldsymbol{\theta}_m|m)p(\boldsymbol{y}|\boldsymbol{\theta}_m, \boldsymbol{d}, m)/Z_m$$

  where $\boldsymbol{y}$ represent the data, $\boldsymbol{d}$ the design, $\boldsymbol{\theta}_m$ the model parameters, $p(\boldsymbol{\theta}_m|m)$ and $p(\boldsymbol{y}|\boldsymbol{\theta}_m, \boldsymbol{d}, m)$ are the prior and the likelihood for model m and model m has prior model probability $p(m)$, for $m = 1, \ldots, K$.

- Z_m is the model evidence, defined as

$$Z_m = p(\boldsymbol{y}|\boldsymbol{d}, m) = \int_{\boldsymbol{\theta}_m} p(\boldsymbol{y}|\boldsymbol{\theta}_m, \boldsymbol{d}, m)p(\boldsymbol{\theta}_m|m)\mathrm{d}\boldsymbol{\theta}_m$$

- Z_m used for model choice (proportional to posterior model probabilities)

Bayesian design

- Find a design $\boldsymbol{d}$ to address particular experimental aims
- Aim is encapsulated in a utility function $u(\boldsymbol{d}, \boldsymbol{y}, m)$
- Could include parameter estimation, model selection and/or prediction
- Maximise expected utility $\boldsymbol{d}^* = \arg\max_{\boldsymbol{d}} u(\boldsymbol{d})$, where

$$u(\boldsymbol{d}) = \sum_{m=1}^{K} p(m) \int_{\boldsymbol{y}} u(\boldsymbol{d}, \boldsymbol{y}, m) p(\boldsymbol{y}|\boldsymbol{d}, m) \mathrm{d}\boldsymbol{y}.$$

- $u(\boldsymbol{d}, \boldsymbol{y}, m)$ is some measure of information gained from $\boldsymbol{d}$ given model m and observed data $\boldsymbol{y}$.

Estimation utilities

- Shannon information gain on θ (Shannon, 1948)

$$u(\boldsymbol{d}, \boldsymbol{y}, m) = \int_{\boldsymbol{\theta}_m} p(\boldsymbol{\theta}_m|m, \boldsymbol{y}, \boldsymbol{d}) \log p(\boldsymbol{y}|\boldsymbol{\theta}_m, m, \boldsymbol{d}) \mathrm{d}\boldsymbol{\theta}_m - \log p(\boldsymbol{y}|m, \boldsymbol{d})$$

- Negative squared loss (Overstall and Woods, 2017)

$$u(\boldsymbol{d}, \boldsymbol{\theta}_m, \boldsymbol{y}, m) = -(\boldsymbol{\theta}_m - E[\boldsymbol{\theta}_m])'(\boldsymbol{\theta}_m - E[\boldsymbol{\theta}_m])$$

where $E[\boldsymbol{\theta}_m] = \sum_{m=1}^{K} p(m) \int_{\boldsymbol{\theta}_m} \boldsymbol{\theta}_m p(\boldsymbol{\theta}_m|\boldsymbol{y}, \boldsymbol{d}, m) \mathrm{d}\boldsymbol{\theta}_m$

- Need $p(\boldsymbol{\theta}_m|m, \boldsymbol{y}, \boldsymbol{d})$ to get $u(\boldsymbol{d}, \boldsymbol{y}, m)$
- Difficult to approximate $\log p(\boldsymbol{y}|m, \boldsymbol{d})$
- Computationally difficult to efficiently approximate $p(\boldsymbol{\theta}_m|m, \boldsymbol{y}, \boldsymbol{d})$ (more details later)

Model discrimination utilities

- Mutual information (Box and Hill, 1967, Drovandi, McGree, Pettitt, 2014)

$$u(\boldsymbol{d}, \boldsymbol{y}, m) = \log p(m|\boldsymbol{y}, \boldsymbol{d})$$

- 01 utility (Overstall, McGree, Drovandi, 2018)

$$u(\boldsymbol{d}, \boldsymbol{y}, m) = I(m = \arg\max_w p(w|\boldsymbol{y}\boldsymbol{d})), w = 1, \ldots, K$$

- Need $p(m|\boldsymbol{y}, \boldsymbol{d})$ to get $u(\boldsymbol{d}, \boldsymbol{y}, m)$
- Computationally difficult

Approximating expected utility

- $u(\boldsymbol{d})$ typically cannot be solved analytically
- Can be approximated using Monte Carlo integration

$$u(\boldsymbol{d}) \approx \sum_{m=1}^{K} p(m) \frac{1}{B} \sum_{b=1}^{B} u(\boldsymbol{d}, \boldsymbol{y}_{mb}, m),$$

  where $\boldsymbol{y}_{mb} \sim p(\boldsymbol{y}|\boldsymbol{\theta}_{mb}, m, \boldsymbol{d})$ and $\boldsymbol{\theta}_{mb} \sim p(\boldsymbol{\theta}_m|m)$.
- BK evaluation of $u(\boldsymbol{d}, \boldsymbol{y}_{mb}, m)$ needed
- Hence, BK posterior distributions need to be approximated or sampled from to approximated $u(\boldsymbol{d})$.
- Computationally challenging task. How can this be achieved efficiently?

Approximating utility

- Long et al. (2013) and Overstall, McGree and Drovandi (2018) used the Laplace approximation for efficiently estimating $u(\boldsymbol{d}, m, \boldsymbol{y})$;
- The main result is that the approximation to the posterior distribution of $\boldsymbol{\theta}_m$ has the following multivariate Normal form:

$$\hat{p}(\boldsymbol{\theta}_m|\boldsymbol{y}, \boldsymbol{d}, m) = (2\pi)^{-\frac{q_m}{2}} |\boldsymbol{\Sigma}_{my}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{\theta}_m - \hat{\boldsymbol{\theta}}_{my})^t \boldsymbol{\Sigma}_{my}^{-1} (\boldsymbol{\theta}_m - \hat{\boldsymbol{\theta}}_{my})\right)$$

where q_m denotes the number of parameters in model m, $\hat{\boldsymbol{\theta}}_{my}$ and $\boldsymbol{\Sigma}_{my}$ denote the posterior mode and posterior variance-covariance matrix, respectively, for model m upon the observation of $\boldsymbol{y}$ from design $\boldsymbol{d}$, for $m = 1, 2, \ldots, K$.

Approximating utility

- For posterior inference on m, the posterior model probability ($p(m|\boldsymbol{y}, \boldsymbol{d})$) can be considered;
- Proportional to the model evidence;
- Based on Laplace approximation, the model evidence can be approximated as follows:

$$\hat{p}(\boldsymbol{y}|m, \boldsymbol{d}) = (2\pi)^{\frac{q_m}{2}} |\boldsymbol{\Sigma}_{my}|^{\frac{1}{2}} p(\boldsymbol{y}|\hat{\boldsymbol{\theta}}_{my}, \boldsymbol{d}) p(\hat{\boldsymbol{\theta}}_{my}|m). \quad (1)$$

- Thus, posterior summaries such as $u(\boldsymbol{d}, m, \boldsymbol{y})$ can be evaluated based on the above Laplace approximation facilitating a relatively efficient approximation to $u(\boldsymbol{d})$;
- However, non-smooth utility function leading to a difficult optimisation problem;
- Try using statistical emulators?

The use of emulators in locating Bayesian designs

- Muller and Parmigiani (1995) - Curve fitting methods to smooth Monte Carlo draws (2D)
- Weaver et al. (2016) - Gaussian process with EQI (3D)
- Jones et al. (2016) - Bayes linear analysis (9D)
- Approximate coordinate exchange (Overstall and Woods, 2017) ($\approx$ 200D);
  - Extension of the coordinate exchange algorithm;
  - Emulator used to interpolate $\hat{u}(\boldsymbol{d})$ in one dimension at a time;
  - Can be inefficient if one-dimensional solution changes depending on the values of other design elements.

The ACE algorithm

- Optimises each design element one-at-a-time
- Instead of exchanging a discrete set of points for each design element, fit an emulator and optimise
- With k explanatory variables, n design points,

$$\boldsymbol{d} = \begin{bmatrix} d_{11} & d_{21} & \dots & d_{k1} \\ d_{12} & d_{22} & \dots & d_{k2} \\ \vdots & & & \\ d_{1n} & d_{2n} & \dots & d_{kn} \end{bmatrix}$$

- Emulate $u(\boldsymbol{d})$ as a function of $\boldsymbol{d}_{ip}$ given $\boldsymbol{d}_{-ij}$ is fixed. Denote as $u(\boldsymbol{d}_{ij}|\boldsymbol{d}_{-ij})$, for $i = 1, \dots, n$ and $j = 1, \dots, k$
- Optimise emulator (brute force)
- Cycle through all nk design points R times

Emulation

- Chosen emulator is the Gaussian Process (GP, Rasmussen and Williams, 2006)
- Defined by a mean and covariance function:

$$f(\boldsymbol{z}) \sim \mathcal{GP}(m(\boldsymbol{z}), k(\boldsymbol{z}, \boldsymbol{\gamma})),$$

where $\boldsymbol{z}$ are predictor variables, mean function $m(\boldsymbol{z})$ is the expected value at $\boldsymbol{z}$ and $k(\boldsymbol{z}, \boldsymbol{\gamma})$ is the covariance function which models the dependence between function values at z_k and z_l, for $k, l = 1, 2, \ldots, T$.

- Each element of the covariance matrix:

$$k(z_k, z_l; \boldsymbol{\gamma}) = \begin{cases} \gamma_0 + \gamma_1 & \text{if } x = 0 \\ \gamma_1 k(z_k, z_l, \gamma_2) & \text{if } x > 0, \end{cases}$$

where x is the Euclidean distance between z_k and z_l, γ_0 is nugget, γ_1 is partial sill and γ_2 is an additional parameter.

The ACE algorithm

- Some popular choices for $k(z_k, z_l, \gamma_2)$ include:

$$k(z_k, z_l, \gamma_2) = \exp(-0.5(x/\gamma_2)^2) \quad \text{Gaussian}$$
$$k(z_k, z_l, \gamma_2) = \exp(-x/\gamma_2) \quad \text{Exponential}$$
$$k(z_k, z_l, \gamma_2) = (1 + \sqrt{5}x\gamma_2 + (5/3)(x/\gamma_2)^2)\exp(-\sqrt{5}x\gamma_2) \quad \text{Matérn.}$$

- Efficient when $f(z)$ is expensive as can interpolate $f(\tilde{z})$ (based on the fitted zero mean GP):

$$f(\tilde{z}) = k(\tilde{z}, \boldsymbol{Z}, \boldsymbol{\gamma})(k(\boldsymbol{Z}, \boldsymbol{\gamma}) + \gamma_0 \boldsymbol{I})^{-1}\boldsymbol{z},$$

where $\boldsymbol{Z}$ denotes training set, $k(\tilde{z}, \boldsymbol{Z}, \boldsymbol{\gamma})$ evaluates the covariances between $\tilde{z}$ and $\boldsymbol{Z}$, and $\boldsymbol{I}$ denotes the identity matrix.

The ACE algorithm

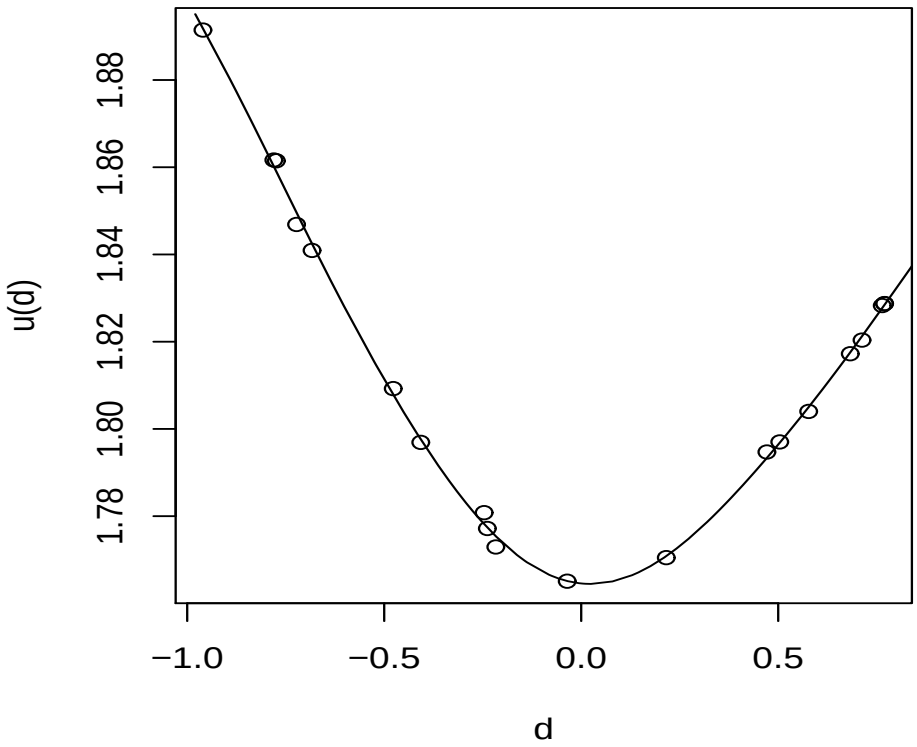

Figure: Emulated utility surface and training points

The k-dimensional ACE algorithm

- Recall

$$\boldsymbol{d} = \begin{bmatrix} \boldsymbol{d}_1 \\ \boldsymbol{d}_2 \\ \vdots \\ \boldsymbol{d}_n \end{bmatrix} = \begin{bmatrix} d_{11} & d_{21} & \dots & d_{k1} \\ d_{12} & d_{22} & \dots & d_{k2} \\ \vdots & & & \\ d_{1n} & d_{2n} & \dots & d_{kn} \end{bmatrix}$$

- Propose optimising $\boldsymbol{d}_i$ simultaneously
- Extend the GP to be k-dimensional
- Emulate $u(\boldsymbol{d})$ as a function of $\boldsymbol{d}_i$ given $\boldsymbol{d}_{-i}$ is fixed. Denote as $u(\boldsymbol{d}_i|\boldsymbol{d}_{-i})$

$$u(\boldsymbol{d}_i|\boldsymbol{d}_{-i}) \sim \mathcal{GP}(\boldsymbol{0}, k(\boldsymbol{D}, \boldsymbol{\gamma})), \quad \boldsymbol{D} \text{ set of training points}$$

- Optimise k-dimensional GP
- Cycle through all n design points R times

The k-dimensional ACE algorithm

Training the emulator

- For each $\boldsymbol{d}_i$, need a training set $\boldsymbol{D}$ to fit GP (for $\hat{u}(\boldsymbol{d}_i|\boldsymbol{d}_{-i})$)
- Consider $\boldsymbol{d}_i$
  - Propose $\boldsymbol{D}$ through random draws from $q(\cdot)$;
  - For each proposal, evaluate $\hat{u}(\boldsymbol{d}_i|\boldsymbol{d}_{-i})$. Denote as $\boldsymbol{u}$
  - Fit GP for $\hat{u}(\boldsymbol{d}_i|\boldsymbol{d}_{-i})$ (via maximum likelihood);
- After training, need to optimise the k-dimensional emulator

The k-dimensional ACE algorithm

Optimising the emulator

- GP is a smooth function, use gradient-based methods;
- Efficient (as above):

$$\tilde{u} = k(\tilde{d}, \boldsymbol{D}, \boldsymbol{\gamma})(k(\boldsymbol{D}, \boldsymbol{\gamma}) + \gamma_0 \boldsymbol{I})^{-1}\boldsymbol{u}. \tag{2}$$

- Once training is complete, loop through the following
  - Optimise the GP based on many random starts
  - Evaluate the 'actual' expected utility $\hat{u}(\boldsymbol{d}_i|\boldsymbol{d}_{-i})$ of maximum
  - Re-fit the GP
- The 'best' design form the above loop is either accepted or rejected
- Then move to next row of the design

The k-dimensional ACE algorithm

Training and optimisation

- Two components of the proposed algorithm; training and optimisation
- Need to consider trade-off in computational resources
- How many utility evaluations for training? How many for optimisation?
- Explored through examples (results omitted, 75% training, 25% optimising).

Choice of covariance function

- k-dimensional emulator, (in general) $k > 1$
- Choice of covariance function is potentially important
- Does this have implications for locating Bayesian designs?
- Explored through examples (some results shown)

Examples

Details of examples

- Two examples to be considered; test problem and motivating example
- Will compare the performance of ACE and k-dimensional ACE
- In both examples, both algorithms will be run in a similar way (equal number of utility evaluations)

Test problem

Logistic regression

- GLMs which are of general interest to the design community (Woods et al., 2006, Dror and Steinberg, 2008)
- Often used to benchmark new computational algorithms in Bayesian inference (Minka, 2001, Cabras et al., 2015)
- Emerging in Bayesian design (Overstall and Woods, 2017, McGree, 2017)
- We consider independent and dependent data settings

Test problem

Logistic regression

- Four factor logistic regression model
- Two scenarios; all data are independent and data are collected in blocks of $n_G = 10$, such that $n = n_G G$, G is total number of blocks
- Model defined as

$$\left(\frac{\pi_{ig}}{1-\pi_{ig}}\right) = \theta_0 + \beta_{0g} + \sum_{j=1}^{k=4}(\theta_j + \beta_{jg})d_{ijg}$$

  where $\boldsymbol{\theta}$ are regression parameters, $\boldsymbol{\beta}$ are block specific parameters, $d_{ij} \in [-1, 1]$, $g = 1, \ldots, G$
- Assumed unknown sign of effect ($\boldsymbol{\theta} = \mathbf{0}$) with variances of 2, $\beta_{jg} \sim N(0, \sigma^2_{jg})$, $\sigma^2_{jg} \sim G(2, 2)$
- For blocked data setting, likelihood approximated (details omitted)

Example 1

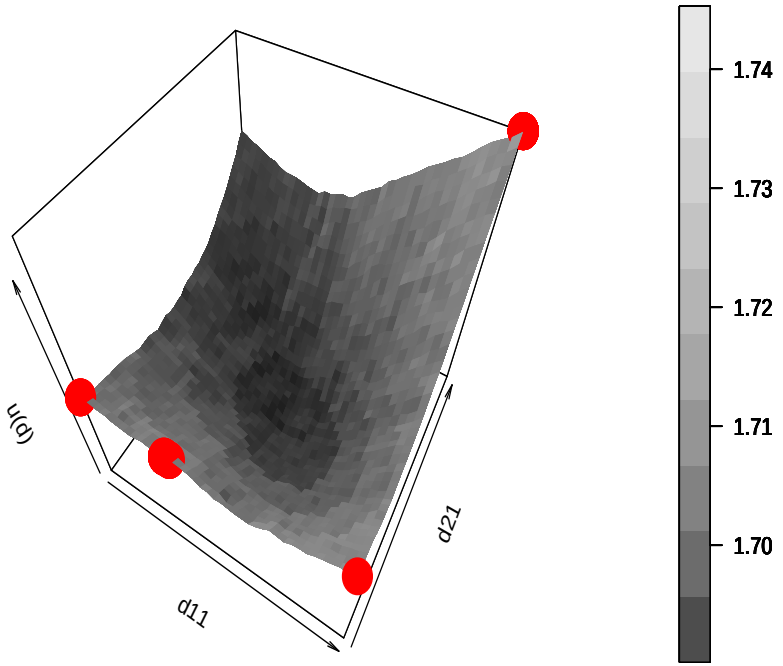

Figure: Utility surface and proposed optimal design points (only in 2D)

Example 1

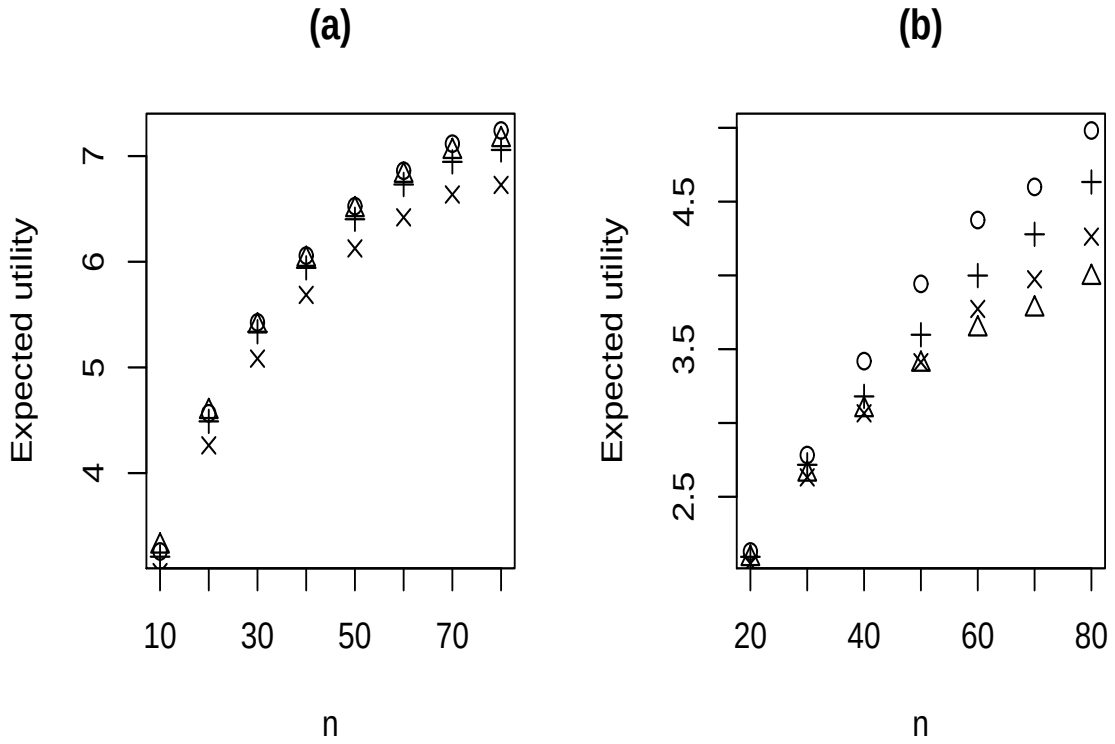

Figure: ACE with exp ($\triangle$), k-ACE with Matérn ($\circ$), exp ($\times$) and quantile improvement ($+$) for Shannon information gain on $\boldsymbol{\theta}$ for standard (a) and hierarchical (b) logistic regression

Motivating example

Screening experiments

- Eight factor logistic regression model
- Unknown if any or all factors are important, 256 competing models
- Model defined as

$$\left(\frac{\pi_i}{1-\pi_i}\right) = \theta_0 + \sum_{j=1}^{k=8} \delta_j \theta_j d_{ij}$$

  where δ_j is a binary indicator for whether factor j is active or not, and $d_{ij} \in [-1, 1]$
- $p(m)$ correct for Bayesian multiplicity (Scott and Berger, 2010)
- Assume uniform priors for $\boldsymbol{\theta}$, lower and upper bounds $(-3, 4, 5, -6, -2.5, -2, -4, -5, -6)$ and $(3, 10, 11, 0, 3.5, 4, 2, 1, 0)$, respectively.

Example 2

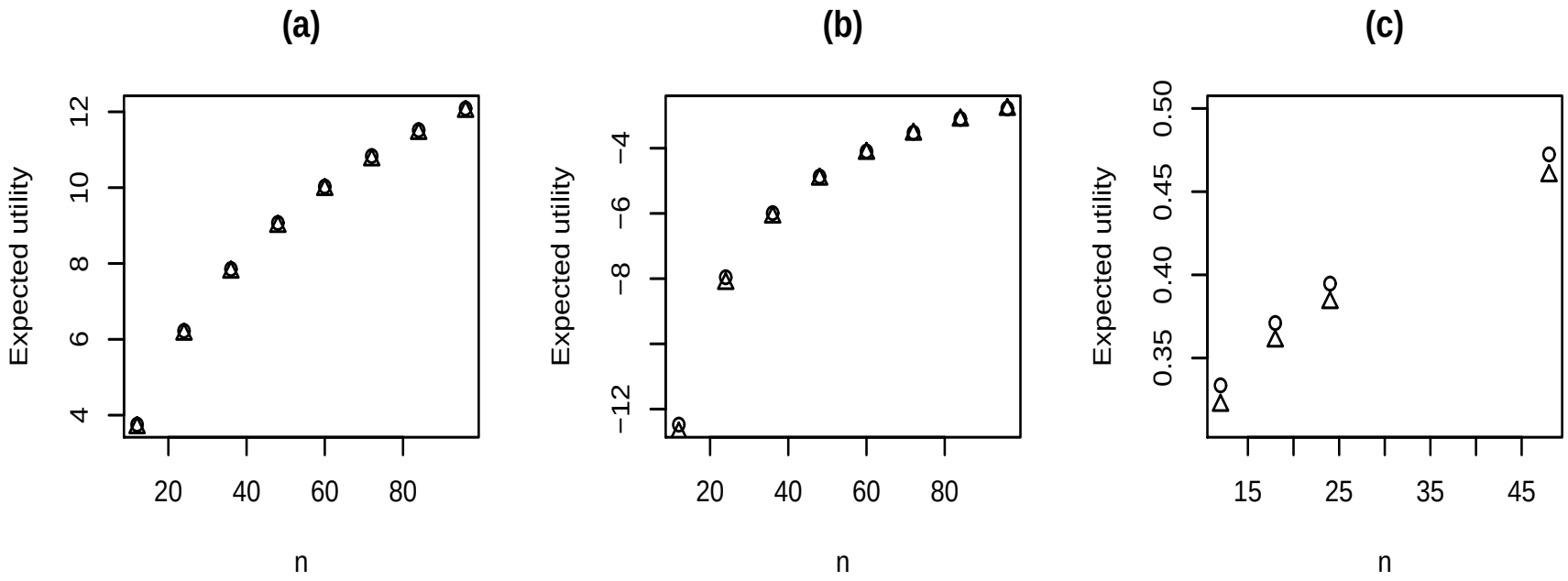

Figure: ACE (△) vs k-ACE (○) for (a) Shannon information gain on θ (b) Negative squared loss and (c) 01-utility

Discussion

- Proposed an extension to the ACE algorithm
- k-dimensional ACE worked better or at least as well as ACE in all examples considered
- Bayesian designs differed based on choice of covariance function - propose using CV to choose
- Also tried quantile improvement for all examples.

Future research

- Potential to consider non-parametric emulators.
- Should avoid some concerns about goodness-of-fit of the emulator
- Could think about more efficient or adaptive proposal distribution for training the emulator
- This is a design problem within itself
- Potential benefits in considering space filling approaches

Selected references

- Box and Hill (1967). *Technometrics*, 9, 57-71.
- Drovandi, McGree and Pettitt (2014). *Journal of Computational and Graphical Statistics*, 23, 3-24.
- McGree (2017). *Computational Statistics & Data Analysis*, 113, 207-225.
- Muller and Parmigiani (1995). *Journal of the American Statistical Association*, 90, 1322-1330.
- Overstall and Woods (2017). *Technometrics*, 59, 458-470.
- Overstall, McGree and Drovandi (2018). *Statistics and Computing*, 28, 343-358.
- Woods, Lewis, Eccleston and Russell (2006) *Technometrics*, 48, 284-292.