Optimal Constrained Design of Control Charts Using Stochastic Approximations

Daniele Zago\textsuperscript{1}  Giovanna Capizzi\textsuperscript{1}  Peihua Qiu\textsuperscript{2}

2023 INFORMS Annual Meeting  October 15, 2023

\textsuperscript{1}University of Padova, Padova, Italy  
\textsuperscript{2}University of Florida, Gainesville, USA
Introduction on Statistical Process Monitoring
Framework

- The goal of SPM is to detect the presence of a **change** in a **sequential process**

\[
X_n \sim \begin{cases} 
F_0(\cdot) & \text{for } n = 0, 1, 2, \ldots, \tau - 1 \quad \text{IN-CONTROL (IC)}, \\
F_1(\cdot) & \text{for } n = \tau, \tau + 1, 2, \ldots \quad \text{OUT-OF-CONTROL (OC)}. 
\end{cases}
\]

- Control charts are the main tools to test the stability of the process using incoming observations.

Control charts

1. **Monitoring statistic:** \( C_n \), e.g. \( C_n = \max \left\{ 0, C_{n-1} + \left( \frac{X_n - \mu_0}{\sigma_0} \right) - k \right\} \).
2. **Control limit:** \( h > 0 \) for all \( n > 0 \).
3. **Run length:** \( RL = \inf \left\{ n : C_n > h \right\} \).
Framework

- The goal of SPM is to detect the presence of a change in a sequential process $X_n \sim \begin{cases} F_0(\cdot) & \text{for } n = 0, 1, 2, \ldots, \tau - 1 \quad \text{IN-CONTROL (IC)}, \\ F_1(\cdot) & \text{for } n = \tau, \tau + 1, 2, \ldots \quad \text{OUT-OF-CONTROL (OC)}. \end{cases}$

- Control charts are the main tools to test the stability of the process using incoming observations.

Control charts

1. Monitoring statistic: $C_n$, e.g. $C_n = \max \left\{ 0, C_{n-1} + \left( \frac{X_n - \mu_0}{\sigma_0} \right) - k \right\}$.
2. Control limit: $h > 0$ for all $n > 0$.
3. Run length: $RL = \inf \{ n : C_n > h \}$. 
Statistical Process Monitoring (SPM) – General framework

Framework

• The goal of SPM is to detect the presence of a **change in a sequential process**

\[ X_n \sim \begin{cases} F_0(\cdot) & \text{for } n = 0, 1, 2, \ldots, \tau - 1 \quad \text{IN-CONTROL (IC)}, \\ F_1(\cdot) & \text{for } n = \tau, \tau + 1, 2, \ldots \quad \text{OUT-OF-CONTROL (OC)}. \end{cases} \]

• Control charts are the main tools to test the stability of the process using incoming observations.

Control charts

1. Monitoring statistic: \( C_n \), e.g. \( C_n = \max \left\{ 0, C_{n-1} + \left( \frac{X_n - \mu_0}{\sigma_0} \right) - k \right\} \).

2. Control limit: \( h > 0 \) for all \( n > 0 \).

3. **Run length**: \( RL = \inf \{ n : C_n > h \} \).
Framework

• The goal of SPM is to detect the presence of a change in a sequential process

\[ X_n \sim \begin{cases} 
F_0(\cdot) & \text{for } n = 0, 1, 2, \ldots, \tau - 1 \quad \text{IN-CONTROL (IC)}, \\
F_1(\cdot) & \text{for } n = \tau, \tau + 1, 2, \ldots \quad \text{OUT-OF-CONTROL (OC)}. 
\end{cases} \]

• Control charts are the main tools to test the stability of the process using incoming observations.

Control charts

1. Monitoring statistic: \( C_n \), e.g. \( C_n = \max \left\{ 0, C_{n-1} + \left( \frac{X_n - \mu_0}{\sigma_0} \right) - k \right\} \).
2. Control limit: \( h > 0 \) for all \( n > 0 \).
3. Run length: \( RL = \inf \{ n : C_n > h \} \).
Statistical Process Monitoring (SPM) – General framework

Framework

• The goal of SPM is to detect the presence of a **change** in a **sequential process**

\[ X_n \sim \begin{cases} 
F_0(\cdot) & \text{for } n = 0, 1, 2, \ldots, \tau - 1 \quad \text{IN-CONTROL (IC)}, \\
F_1(\cdot) & \text{for } n = \tau, \tau + 1, 2, \ldots \quad \text{OUT-OF-CONTROL (OC)}. 
\end{cases} \]

• Control charts are the main tools to test the stability of the process using incoming observations.

Control charts

1. Monitoring statistic: \( C_n \), e.g. \( C_n = \max \left\{ 0, C_{n-1} + \left( \frac{X_n - \mu_0}{\sigma_0} \right) - k \right\} \).
2. Control limit: \( h > 0 \) for all \( n > 0 \).
3. **Run length**: \( RL = \inf \{ n : C_n > h \} \).
Design of a control chart

• Selection of control chart tuning parameters, which typically depend on the expected process shift.

• Selection of the control limit $h$, which is usually chosen so that

$$\text{ARL}_{IC} = \mathbb{E}[RL|\tau = \infty] = \text{ARL}_0,$$

for some value of $\text{ARL}_0$ (e.g. 200, 370, 500, …)

Performance metrics

• Small values of $\text{ARL}_{\tau_0} = \mathbb{E}[RL|\tau = \tau_0]$ mean that the chart performs better.
Design of a control chart

- Selection of control chart tuning parameters, which typically depend on the expected process shift.
- Selection of the control limit $h$, which is usually chosen so that

$$\text{ARL}_{IC} = \mathbb{E}[RL | \tau = \infty] = \text{ARL}_0,$$

for some value of $\text{ARL}_0$ (e.g. 200, 370, 500, ...)

Performance metrics

- Small values of $\text{ARL}_{\tau_0} = \mathbb{E}[RL | \tau = \tau_0]$ mean that the chart performs better.
Design of a control chart

- Selection of control chart tuning parameters, which typically depend on the expected process shift.
- Selection of the control limit $h$, which is usually chosen so that

$$\text{ARL}_{IC} = \mathbb{E}[RL|\tau = \infty] = \text{ARL}_0,$$

for some value of $\text{ARL}_0$ (e.g. 200, 370, 500, …)

Performance metrics

- Small values of $\text{ARL}_{\tau_0} = \mathbb{E}[RL|\tau = \tau_0]$ mean that the chart performs better.
Example of (analytical) optimization

• Consider a CUSUM control chart $C_n = \max\left\{ 0, C_{n-1} + \left( \frac{X_n - \mu_0}{\sigma_0} \right) - k \right\}$ for monitoring changes in the mean of the process.

• Assume that the IC process observations are i.i.d. $X_n \sim \mathcal{N}(\mu_0, \sigma_0)$.

• Suppose that the expected OC mean is $\mu_1$ and the (standardized) change to be detected is $\delta = \frac{\mu_1 - \mu_0}{\sigma_0}$.

• Then, it is well-known that the tuning parameter that minimizes the $\text{ARL}_1$ is $k = \delta / 2$, irrespective of the value of $\text{ARL}_0$. 
Example of (analytical) optimization

- Consider a CUSUM control chart \( C_n = \max \left\{ 0, C_{n-1} + \left( \frac{X_n - \mu_0}{\sigma_0} \right) - k \right\} \) for monitoring changes in the mean of the process.

- Assume that the IC process observations are i.i.d. \( X_n \sim \mathcal{N}(\mu_0, \sigma_0) \).

- Suppose that the expected OC mean is \( \mu_1 \) and the (standardized) change to be detected is \( \delta = \frac{\mu_1 - \mu_0}{\sigma_0} \).

- Then, it is well-known that the tuning parameter that minimizes the ARL\(_1\) is \( k = \delta / 2 \), irrespective of the value of ARL\(_0\).
Example of (analytical) optimization

• Consider a CUSUM control chart $C_n = \max \left\{ 0, C_{n-1} + \left( \frac{X_n - \mu_0}{\sigma_0} \right) - k \right\}$ for monitoring changes in the mean of the process.

• Assume that the IC process observations are i.i.d. $X_n \sim N(\mu_0, \sigma_0)$.

• Suppose that the expected OC mean is $\mu_1$ and the (standardized) change to be detected is $\delta = \frac{\mu_1 - \mu_0}{\sigma_0}$.

• Then, it is well-known that the tuning parameter that minimizes the ARL$_1$ is $k = \delta / 2$, irrespective of the value of ARL$_0$. 
Example of (analytical) optimization

• Consider a CUSUM control chart $C_n = \max \left\{ 0, C_{n-1} + \left( \frac{X_n - \mu_0}{\sigma_0} \right) - k \right\}$ for monitoring changes in the mean of the process.

• Assume that the IC process observations are i.i.d. $X_n \sim \mathcal{N}(\mu_0, \sigma_0)$.

• Suppose that the expected OC mean is $\mu_1$ and the (standardized) change to be detected is $\delta = \frac{\mu_1 - \mu_0}{\sigma_0}$.

• Then, it is well-known that the tuning parameter that minimizes the $\text{ARL}_1$ is $k = \delta/2$, irrespective of the value of $\text{ARL}_0$. 
Optimization Using Stochastic Approximations
Chart design

- Control charts (and their run lengths) depend on a set of tuning parameters $\zeta \in \mathbb{R}^d$.
- Different values of tuning parameters allow better detection of different magnitudes of parameter shifts.
- The goal is to optimize the $\text{ARL}_{\text{OC}}$ under the constraint on the in-control ARL,

$$
\zeta^* = \arg\min_{\zeta \in \mathcal{Z}} \mathbb{E}_1[\text{RL}(\zeta, h(\zeta))] \\
\text{s.t. } \mathbb{E}_0[\text{RL}(\zeta, h(\zeta))] = \text{ARL}_0,
$$
Chart design

- Control charts (and their run lengths) depend on a set of tuning parameters $\zeta \in \mathbb{R}^d$.
- Different values of tuning parameters allow better detection of different magnitudes of parameter shifts.
- The goal is to optimize the $\text{ARL}_{OC}$ under the constraint on the in-control ARL,

$$
\zeta^* = \arg\min_{\zeta \in \mathcal{Z}} \mathbb{E}_1[RL(\zeta, h(\zeta))] \\
\text{s.t. } \mathbb{E}_0[RL(\zeta, h(\zeta))] = \text{ARL}_0,
$$
Optimal control chart design

Chart design

- Control charts (and their run lengths) depend on a set of tuning parameters $\zeta \in \mathbb{R}^d$.
- Different values of tuning parameters allow better detection of different magnitudes of parameter shifts.
- The goal is to optimize the $\text{ARL}_{\text{OC}}$ under the constraint on the in-control ARL,

$$
\zeta^* = \arg\min_{\zeta \in \mathbb{Z}} \mathbb{E}_1[\text{RL}(\zeta, h(\zeta))]
$$

s.t. $\mathbb{E}_0[\text{RL}(\zeta, h(\zeta))] = \text{ARL}_0,$
Classical methods for optimization

Analytical methods

• Applicable only in very simple cases (CUSUM with Gaussian observations, ...)

Numerical methods

• Numerical quadrature (for example, Knoth, 2017).
• Limitations: applicable for some specific control charts, scales poorly when \( d > 1 \).

Monte-Carlo approaches

• Estimate \( \text{ARL}_{OC} \) for given \( \zeta \) with a large number of simulations and use optimization tools.
  • Grid search (Qiu, 2008; Qiu and Xie, 2021).
  • Other numerical solvers (Capizzi and Masarotto, 2003; Mahmoud and Zahran, 2010).
• Limitations: function is treated as deterministic, methodologies are expensive to scale for \( d > 1 \) (as we will see later).
Classical methods for optimization

Analytical methods

• Applicable only in very simple cases (CUSUM with Gaussian observations, ...)

Numerical methods

• Numerical quadrature (for example, Knoth, 2017).

• Limitations: applicable for some specific control charts, scales poorly when $d > 1$.

Monte-Carlo approaches

• Estimate $\text{ARL}_{OC}$ for given $\zeta$ with a large number of simulations and use optimization tools.
  • Grid search (Qiu, 2008; Qiu and Xie, 2021).
  • Other numerical solvers (Capizzi and Masarotto, 2003; Mahmoud and Zahran, 2010).

• Limitations: function is treated as deterministic, methodologies are expensive to scale for $d > 1$ (as we will see later).
Classical methods for optimization

Analytical methods

• Applicable only in very simple cases (CUSUM with Gaussian observations, ...)

Numerical methods

• Numerical quadrature (for example, Knoth, 2017).

• Limitations: applicable for some specific control charts, scales poorly when $d > 1$.

Monte-Carlo approaches

• Estimate $\text{ARL}_{OC}$ for given $\zeta$ with a large number of simulations and use optimization tools.
  • Grid search (Qiu, 2008; Qiu and Xie, 2021).
  • Other numerical solvers (Capizzi and Masarotto, 2003; Mahmoud and Zahran, 2010).

• Limitations: function is treated as deterministic, methodologies are expensive to scale for $d > 1$ (as we will see later).
Classical methods for optimization

Analytical methods

• Applicable only in very simple cases (CUSUM with Gaussian observations, ...)

Numerical methods

• Numerical quadrature (for example, Knoth, 2017).
• **Limitations**: applicable for some specific control charts, scales poorly when $d > 1$.

Monte-Carlo approaches

• Estimate $\text{ARL}_{OC}$ for given $\zeta$ with a large number of simulations and use optimization tools.
  • Grid search (Qiu, 2008; Qiu and Xie, 2021).
  • Other numerical solvers (Capizzi and Masarotto, 2003; Mahmoud and Zahran, 2010).
• **Limitations**: function is treated as deterministic, methodologies are expensive to scale for $d > 1$ (as we will see later).
Classical methods for optimization

Analytical methods

• Applicable only in very simple cases (CUSUM with Gaussian observations, ...)

Numerical methods

• Numerical quadrature (for example, Knoth, 2017).
• Limitations: applicable for some specific control charts, scales poorly when $d > 1$.

Monte-Carlo approaches

• Estimate $\text{ARL}_{\text{OC}}$ for given $\zeta$ with a large number of simulations and use optimization tools.
  • Grid search (Qiu, 2008; Qiu and Xie, 2021).
  • Other numerical solvers (Capizzi and Masarotto, 2003; Mahmoud and Zahran, 2010).
• Limitations: function is treated as deterministic, methodologies are expensive to scale for $d > 1$ (as we will see later).
A primer on Stochastic Approximations (SA)

Starting point

• Let $Q(\zeta, h(\zeta))$ be the **noisy** function we want to minimize

• Let $\Psi : \mathbb{R}^d \rightarrow \mathcal{Z}$ be the projection onto the nearest point in $\mathcal{Z}$.

• We would like to find the minimum of $Q$ using a gradient descent iteration (Spall, 2003)

$$
\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \frac{\partial Q(\zeta, h(\zeta))}{\partial \zeta} \bigg|_{\zeta=\hat{\zeta}} \right), \quad k = 1, 2, \ldots
$$

• **Problem**: The gradient in (1) is **unknown** and cannot be expressed in closed form.

Stochastic approximations approach

• Estimate the gradient using finite differences and use $\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \tilde{g}_k(\hat{\zeta}_k) \right)$,

• When $\zeta \in \mathbb{R}^d$, the SA method requires evaluating the function $Q$ at $2d$ parameter values.

• **Problem**: Because of the ARL\textsubscript{IC} constraint, the function $Q$ is **expensive** to evaluate.
A primer on Stochastic Approximations (SA)

Starting point

- Let $Q(\zeta, h(\zeta))$ be the \textbf{noisy} function we want to minimize
- Let $\Psi : \mathbb{R}^d \rightarrow \mathbb{Z}$ be the projection onto the nearest point in $\mathbb{Z}$.
  - We would like to find the minimum of $Q$ using a gradient descent iteration (Spall, 2003)
    \[
    \hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \frac{\partial Q(\zeta, h(\zeta))}{\partial \zeta} \bigg|_{\zeta=\hat{\zeta}} \right), \quad k = 1, 2, \ldots \tag{1}
    \]
  - \textbf{Problem:} The gradient in (1) is \textbf{unknown} and cannot be expressed in closed form.

Stochastic approximations approach

- Estimate the gradient using finite differences and use $\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \hat{g}_k(\hat{\zeta}_k) \right)$,
- When $\zeta \in \mathbb{R}^d$, the SA method requires evaluating the function $Q$ at $2d$ parameter values.
- \textbf{Problem:} Because of the ARL$_{IC}$ constraint, the function $Q$ is \textbf{expensive} to evaluate.
A primer on Stochastic Approximations (SA)

Starting point

• Let $Q(\zeta, h(\zeta))$ be the noisy function we want to minimize
• Let $\Psi : \mathbb{R}^d \rightarrow \mathcal{Z}$ be the projection onto the nearest point in $\mathcal{Z}$.
• We would like to find the minimum of $Q$ using a gradient descent iteration (Spall, 2003)

$$
\tilde{\zeta}_{k+1} = \Psi \left( \tilde{\zeta}_k - a_k \frac{\partial Q(\zeta, h(\zeta))}{\partial \zeta} \bigg|_{\zeta=\tilde{\zeta}} \right), \quad k = 1, 2, \ldots
$$

• **Problem:** The gradient in (1) is unknown and cannot be expressed in closed form.

Stochastic approximations approach

• Estimate the gradient using finite differences and use $\tilde{\zeta}_{k+1} = \Psi \left( \tilde{\zeta}_k - a_k \hat{g}_k(\tilde{\zeta}_k) \right)$,
• When $\zeta \in \mathbb{R}^d$, the SA method requires evaluating the function $Q$ at $2d$ parameter values.
• **Problem:** Because of the ARL$_{IC}$ constraint, the function $Q$ is expensive to evaluate.
A primer on Stochastic Approximations (SA)

Starting point

• Let $Q(\zeta, h(\zeta))$ be the **noisy** function we want to minimize
• Let $\Psi : \mathbb{R}^d \rightarrow \mathcal{Z}$ be the projection onto the nearest point in $\mathcal{Z}$.
• We would like to find the minimum of $Q$ using a gradient descent iteration (Spall, 2003)

$$\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \left. \frac{\partial Q(\zeta, h(\zeta))}{\partial \zeta} \right|_{\zeta=\hat{\zeta}} \right), \quad k = 1, 2, \ldots$$

(1)

• **Problem:** The gradient in (1) is **unknown** and cannot be expressed in closed form.

Stochastic approximations approach

• Estimate the gradient using finite differences and use $\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \hat{g}_k(\hat{\zeta}_k) \right)$,
• When $\zeta \in \mathbb{R}^d$, the SA method requires evaluating the function $Q$ at $2d$ parameter values.
• **Problem:** Because of the ARL$_{IC}$ constraint, the function $Q$ is **expensive** to evaluate.
A primer on Stochastic Approximations (SA)

Starting point

• Let $Q(\zeta, h(\zeta))$ be the **noisy** function we want to minimize
• Let $\Psi : \mathbb{R}^d \rightarrow \mathbb{Z}$ be the projection onto the nearest point in $\mathbb{Z}$.
• We would like to find the minimum of $Q$ using a gradient descent iteration (Spall, 2003)

\[
\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \frac{\partial Q(\zeta, h(\zeta))}{\partial \zeta} \bigg|_{\zeta=\hat{\zeta}} \right), \quad k = 1, 2, \ldots
\]

• **Problem**: The gradient in (1) is **unknown** and cannot be expressed in closed form.

Stochastic approximations approach

• Estimate the gradient using finite differences and use $\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \hat{g}_k(\zeta) \right)$,
• When $\zeta \in \mathbb{R}^d$, the SA method requires evaluating the function $Q$ at $2d$ parameter values.
• **Problem**: Because of the ARL$_{IC}$ constraint, the function $Q$ is **expensive** to evaluate.
A primer on Stochastic Approximations (SA)

Starting point

• Let $Q(\zeta, h(\zeta))$ be the noisy function we want to minimize
• Let $\Psi : \mathbb{R}^d \rightarrow \mathbb{Z}$ be the projection onto the nearest point in $\mathbb{Z}$.
• We would like to find the minimum of $Q$ using a gradient descent iteration (Spall, 2003)

$$\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \frac{\partial Q(\zeta, h(\zeta))}{\partial \zeta} \bigg|_{\zeta=\hat{\zeta}} \right), \quad k = 1, 2, \ldots \quad (1)$$

• Problem: The gradient in (1) is unknown and cannot be expressed in closed form.

Stochastic approximations approach

• Estimate the gradient using finite differences and use $\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \hat{g}_k(\hat{\zeta}_k) \right)$,
• When $\zeta \in \mathbb{R}^d$, the SA method requires evaluating the function $Q$ at $2d$ parameter values.
• Problem: Because of the ARLIC constraint, the function $Q$ is expensive to evaluate.
A primer on Stochastic Approximations (SA)

Starting point

• Let $Q(\zeta, h(\zeta))$ be the noisy function we want to minimize
• Let $\Psi : \mathbb{R}^d \rightarrow \mathcal{Z}$ be the projection onto the nearest point in $\mathcal{Z}$.
• We would like to find the minimum of $Q$ using a gradient descent iteration (Spall, 2003)

$$\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \frac{\partial Q(\zeta, h(\zeta))}{\partial \zeta} \bigg|_{\zeta=\hat{\zeta}} \right), \quad k = 1, 2, \ldots$$  (1)

• **Problem**: The gradient in (1) is unknown and cannot be expressed in closed form.

Stochastic approximations approach

• Estimate the gradient using finite differences and use $\hat{\zeta}_{k+1} = \Psi \left( \hat{\zeta}_k - a_k \hat{g}_k(\hat{\zeta}_k) \right)$,
• When $\zeta \in \mathbb{R}^d$, the SA method requires evaluating the function $Q$ at $2d$ parameter values.
• **Problem**: Because of the ARL$_{IC}$ constraint, the function $Q$ is expensive to evaluate.
Simultaneous Perturbation Stochastic Approximations (SPSA)

Simultaneous perturbations approach

- Let $\Delta_k = (\Delta_{1k}, \ldots, \Delta_{dk})$ be independent zero-mean random variables that are symmetric.
- Typically (Spall, 2003)
  \[ \Delta_{jk} \overset{iid}{\sim} \begin{cases} 1 & \text{with probability } 1/2, \\ -1 & \text{with probability } 1/2. \end{cases} \]
- Perturb the current parameter estimates: $\hat{\zeta}_k^+ = \Psi(\hat{\zeta}_k + c_k \Delta_k)$ and $\hat{\zeta}_k^- = \Psi(\hat{\zeta}_k - c_k \Delta_k)$.
- Gradient estimate is
  \[ \hat{g}_k(\hat{\zeta}_k) = \frac{Q(\hat{\zeta}_k^+, h(\hat{\zeta}_k^+)) - Q(\hat{\zeta}_k^-, h(\hat{\zeta}_k^-))}{2c_k} \begin{pmatrix} \Delta_{1k}^{-1} \\ \vdots \\ \Delta_{dk}^{-1} \end{pmatrix}, \]  
  \[ (2) \]
- **Advantage**: requires 2 evaluations of $Q$ irrespective of the dimension $d$. 
Simultaneous Perturbation Stochastic Approximations (SPSA)

Simultaneous perturbations approach

• Let $\Delta_k = (\Delta_{1k}, \ldots, \Delta_{dk})$ be independent zero-mean random variables that are symmetric.
• Typically (Spall, 2003)

\[
\Delta_{jk} \sim \begin{cases} 
1 & \text{with probability } 1/2, \\
-1 & \text{with probability } 1/2.
\end{cases}
\]

• Perturb the current parameter estimates: $\hat{\zeta}_k^+ = \Psi(\hat{\zeta}_k + c_k \Delta_k)$ and $\hat{\zeta}_k^- = \Psi(\hat{\zeta}_k - c_k \Delta_k)$.
• Gradient estimate is

\[
\hat{g}_k(\hat{\zeta}_k) = \frac{Q(\hat{\zeta}_k^+, h(\hat{\zeta}_k^+)) - Q(\hat{\zeta}_k^-, h(\hat{\zeta}_k^-))}{2c_k} \begin{pmatrix} \Delta^{-1}_{1k} \\ \vdots \\ \Delta^{-1}_{dk} \end{pmatrix},
\]

• Advantage: requires 2 evaluations of $Q$ irrespective of the dimension $d$. 
Simultaneous Perturbation Stochastic Approximations (SPSA)

Simultaneous perturbations approach

• Let $\Delta_k = (\Delta_{1k}, \ldots, \Delta_{dk})$ be independent zero-mean random variables that are symmetric.
• Typically (Spall, 2003)

\[
\Delta_{jk} \sim \begin{cases} 
1 & \text{with probability } 1/2, \\
-1 & \text{with probability } 1/2.
\end{cases}
\]

• Perturb the current parameter estimates: $\hat{\zeta}_k^+ = \Psi(\hat{\zeta}_k + c_k \Delta_k)$ and $\hat{\zeta}_k^- = \Psi(\hat{\zeta}_k - c_k \Delta_k)$.
• Gradient estimate is

\[
\hat{g}_k(\hat{\zeta}_k) = \frac{Q(\hat{\zeta}_k^+, h(\hat{\zeta}_k^+)) - Q(\hat{\zeta}_k^-, h(\hat{\zeta}_k^-))}{2c_k} \begin{pmatrix} 
\Delta_{1k}^{-1} \\
\vdots \\
\Delta_{dk}^{-1}
\end{pmatrix},
\]

(2)

• Advantage: requires 2 evaluations of $Q$ irrespective of the dimension $d$. 

Simultaneous Perturbation Stochastic Approximations (SPSA)

Simultaneous perturbations approach

• Let $\Delta_k = (\Delta_{1k}, \ldots, \Delta_{dk})$ be independent zero-mean random variables that are symmetric.

• Typically (Spall, 2003)

$$\Delta_{jk} \overset{\text{iid}}{\sim} \begin{cases} 1 & \text{with probability } 1/2, \\ -1 & \text{with probability } 1/2. \end{cases}$$

• Perturb the current parameter estimates: $\hat{\zeta}_k^+ = \Psi(\hat{\zeta}_k + c_k \Delta_k)$ and $\hat{\zeta}_k^- = \Psi(\hat{\zeta}_k - c_k \Delta_k)$.

• Gradient estimate is

$$\hat{g}_k(\hat{\zeta}_k) = \frac{Q(\hat{\zeta}_k^+, h(\hat{\zeta}_k^+)) - Q(\hat{\zeta}_k^-, h(\hat{\zeta}_k^-))}{2c_k} \begin{pmatrix} \Delta_{1k}^{-1} \\ \vdots \\ \Delta_{dk}^{-1} \end{pmatrix},$$

• Advantage: requires 2 evaluations of $Q$ irrespective of the dimension $d$. 
Simultaneous Perturbation Stochastic Approximations (SPSA)

Simultaneous perturbations approach

- Let $\Delta_k = (\Delta_{1k}, \ldots, \Delta_{dk})$ be independent zero-mean random variables that are symmetric.
- Typically (Spall, 2003)
  \[
  \Delta_{jk} \sim \begin{cases} 
  1 & \text{with probability } 1/2, \\
  -1 & \text{with probability } 1/2.
  \end{cases}
  \]

- Perturb the current parameter estimates: $\hat{\zeta}_k^+ = \Psi(\hat{\zeta}_k + c_k \Delta_k)$ and $\hat{\zeta}_k^- = \Psi(\hat{\zeta}_k - c_k \Delta_k)$.
- Gradient estimate is
  \[
  \hat{g}_k(\hat{\zeta}_k) = \frac{Q(\hat{\zeta}_k^+, h(\hat{\zeta}_k^+)) - Q(\hat{\zeta}_k^-, h(\hat{\zeta}_k^-))}{2c_k} \begin{pmatrix} 
  \Delta_{1k}^{-1} \\
  \vdots \\
  \Delta_{dk}^{-1}
  \end{pmatrix},
  \]
  \hspace{1cm} (2)

- **Advantage**: requires 2 evaluations of $Q$ irrespective of the dimension $d$. 


Further enhancements

Polyak averaging

- The estimate of the optimum $\zeta^*$ at iteration $k$ is the average over the optimization path,

$$\bar{\zeta}_k = \frac{1}{k - N_t} \sum_{\ell=N_t+1}^{k} \hat{\zeta}_\ell,$$

- Averaging increases stability while providing a similar convergence rate to the solution.

Noise reduction

- Once $h(\hat{\zeta}_k)$ is found, $r = 100$ simulations of $Q(\hat{\zeta}_k)$ are used to estimate $\hat{g}(\hat{\zeta}_k)$.

Computational bottleneck

- Calculation of the control limits $h(\hat{\zeta}_k^+)$ and $h(\hat{\zeta}_k^-)$ is the algorithm's bottleneck.
- We use a low-precision SA algorithm (Capizzi and Masarotto, 2016) with a warm-start initialization that allows it to become more accurate as $k$ increases.
Further enhancements

Polyak averaging

• The estimate of the optimum $\zeta^*$ at iteration $k$ is the average over the optimization path,

$$\bar{\zeta}_k = \frac{1}{k - N_f} \sum_{\ell=N_f+1}^{k} \tilde{\zeta}_\ell,$$

• Averaging increases stability while providing a similar convergence rate to the solution.

Noise reduction

• Once $h(\hat{\zeta}_k)$ is found, $r = 100$ simulations of $Q(\hat{\zeta}_k)$ are used to estimate $\hat{g}(\hat{\zeta}_k)$.

Computational bottleneck

• Calculation of the control limits $h(\hat{\zeta}_k^+)$ and $h(\hat{\zeta}_k^-)$ is the algorithm's bottleneck.

• We use a low-precision SA algorithm (Capizzi and Masarotto, 2016) with a warm-start initialization that allows it to become more accurate as $k$ increases.
Further enhancements

Polyak averaging

- The estimate of the optimum $\zeta^*$ at iteration $k$ is the average over the optimization path,

$$\bar{\zeta}_k = \frac{1}{k - N_t} \sum_{\ell=N_t+1}^{k} \hat{\zeta}_\ell,$$

- Averaging increases stability while providing a similar convergence rate to the solution.

Noise reduction

- Once $h(\hat{\zeta}_k)$ is found, $r = 100$ simulations of $Q(\hat{\zeta}_k)$ are used to estimate $\hat{g}(\hat{\zeta}_k)$.

Computational bottleneck

- Calculation of the control limits $h(\hat{\zeta}_k^+)$ and $h(\hat{\zeta}_k^-)$ is the algorithm's bottleneck.
- We use a low-precision SA algorithm (Capizzi and Masarotto, 2016) with a warm-start initialization that allows it to become more accurate as $k$ increases.
Further enhancements

Polyak averaging

• The estimate of the optimum $\zeta^*$ at iteration $k$ is the average over the optimization path,

$$\bar{\zeta}_k = \frac{1}{k - N_t} \sum_{\ell=N_t+1}^{k} \hat{\zeta}_\ell,$$

• Averaging increases stability while providing a similar convergence rate to the solution.

Noise reduction

• Once $h(\hat{\zeta}_k)$ is found, $r = 100$ simulations of $Q(\hat{\zeta}_k)$ are used to estimate $\hat{g}(\hat{\zeta}_k)$.

Computational bottleneck

• Calculation of the control limits $h(\hat{\zeta}_k^+)$ and $h(\hat{\zeta}_k^-)$ is the algorithm's bottleneck.

• We use a low-precision SA algorithm (Capizzi and Masarotto, 2016) with a warm-start initialization that allows it to become more accurate as $k$ increases.
Further enhancements

Polyak averaging

- The estimate of the optimum $\zeta^*$ at iteration $k$ is the average over the optimization path,

$$\bar{\zeta}_k = \frac{1}{k - N_t} \sum_{\ell=N_t+1}^{k} \hat{\zeta}_\ell,$$

- Averaging increases stability while providing a similar convergence rate to the solution.

Noise reduction

- Once $h(\hat{\zeta}_k)$ is found, $r = 100$ simulations of $Q(\hat{\zeta}_k)$ are used to estimate $\hat{g}(\hat{\zeta}_k)$.

Computational bottleneck

- Calculation of the control limits $h(\hat{\zeta}_k^+)$ and $h(\hat{\zeta}_k^-)$ is the algorithm's bottleneck.
- We use a low-precision SA algorithm (Capizzi and Masarotto, 2016) with a warm-start initialization that allows it to become more accurate as $k$ increases.
Convergence criterion

“Stochastic” convergence criterion

- A reasonable stopping rule of the algorithm is $|\mathbb{E}[\hat{g}_{jk}(\zeta_k)]| \leq \nu$, for a small value of $\nu$
- We leverage the asymptotic distribution of the PR averaging scheme (Maryak, 1997),
  \[ k^{1/3} \left[ Q'(\tilde{\zeta}_k) - Q'(\zeta^* - \mu) \right] \sim N_d(0, Q''(\zeta^* - \mu)\Sigma Q''(\zeta^* - \mu)^T) \]
- Using a similar approach to Capizzi and Masarotto (2016), a stopping criterion can be defined as
  \[ \bar{N}_s = \inf \left\{ k > N_m + N_f : k \geq \left( \frac{Z}{\nu} \right)^2 \max_{j=1, \ldots, p} \frac{1}{N - N_f} \sum_{\ell=N_f+1}^{k} \bar{g}_{j\ell}(\zeta_k)^2 \right\} , \]
- $z$ is the $[(1 - \nu)/2]$-th quantile of the standard normal distribution.
- $N_m + N_f$ is specified to avoid a premature ending of the algorithm.
Convergence criterion

“Stochastic” convergence criterion

• A reasonable stopping rule of the algorithm is $|\mathbb{E}[\hat{g}_{jk}(\hat{\zeta}_k)]| \leq \nu$, for a small value of $\nu$

• We leverage the asymptotic distribution of the PR averaging scheme (Maryak, 1997),

$$k^{1/3} \left[ Q'(\tilde{\zeta}_k) - Q'(\zeta^* - \mu) \right] \sim N_d(0, Q''(\zeta^* - \mu)\Sigma Q''(\zeta^* - \mu)^T)$$

• Using a similar approach to Capizzi and Masarotto (2016), a stopping criterion can be defined as

$$\tilde{N}_s = \inf \left\{ k > N_m + N_f : k \geq \left( \frac{z}{\nu} \right)^2 \max_{j=1,\ldots,p} \frac{1}{N - N_f} \sum_{\ell=N_f+1}^k \frac{g_{j \ell}(\hat{\zeta}_k)^2}{1} \right\},$$

• $z$ is the $[(1 - \nu)/2]$-th quantile of the standard normal distribution.

• $N_m + N_f$ is specified to avoid a premature ending of the algorithm.
“Stochastic” convergence criterion

• A reasonable stopping rule of the algorithm is $|E\{\hat{g}_{jk}(\hat{\zeta}_k)\}| \leq \nu$, for a small value of $\nu$
• We leverage the asymptotic distribution of the PR averaging scheme (Maryak, 1997),

$$k^{1/3}\left[Q'(\tilde{\zeta}_k) - Q'(\zeta^* - \mu)\right] \sim N_d(0, Q''(\zeta^* - \mu)\Sigma Q''(\zeta^* - \mu)^T)$$

• Using a similar approach to Capizzi and Masarotto (2016), a stopping criterion can be defined as

$$\bar{N}_s = \inf \left\{ k > N_m + N_f : k \geq \left(\frac{Z}{\nu}\right)^2 \max_{j=1,\ldots,p} \frac{1}{N - N_f} \sum_{\ell=N_f+1}^{k} \bar{g}_{j\ell}(\hat{\zeta}_k)^2 \right\},$$

• $z$ is the $[(1 - \nu)/2]$-th quantile of the standard normal distribution.
• $N_m + N_f$ is specified to avoid a premature ending of the algorithm.
“Stochastic” convergence criterion

- A reasonable stopping rule of the algorithm is \( |\mathbb{E}[\hat{g}_{jk}(\hat{\zeta}_k)]| \leq \nu \), for a small value of \( \nu \).
- We leverage the asymptotic distribution of the PR averaging scheme (Maryak, 1997),

\[
 k^{1/3} \left[ Q'(\tilde{\zeta}_k) - Q'(\zeta^* - \mu) \right] \sim N_d(0, Q''(\zeta^* - \mu)\Sigma Q''(\zeta^* - \mu)^T)
\]

- Using a similar approach to Capizzi and Masarotto (2016), a stopping criterion can be defined as

\[
 \bar{N}_s = \inf \left\{ k > N_m + N_f : k \geq \left( \frac{Z}{\nu} \right)^2 \max_{j=1,\ldots,p} \frac{1}{N-N_f} \sum_{\ell=N_f+1}^{k} \bar{g}_{j\ell}(\hat{\zeta}_k)^2 \right\},
\]

- \( z \) is the \([(1 - \nu)/2]\)-th quantile of the standard normal distribution.
- \( N_m + N_f \) is specified to avoid a premature ending of the algorithm.
“Stochastic” convergence criterion

• A reasonable stopping rule of the algorithm is $|\mathbb{E}[\hat{g}_{jk}(\hat{\zeta}_k)]| \leq \nu$, for a small value of $\nu$

• We leverage the asymptotic distribution of the PR averaging scheme (Maryak, 1997),

$$k^{1/3} \left[ Q'(\tilde{\zeta}_k) - Q'(\zeta^* - \mu) \right] \sim N_d(0, Q''(\zeta^* - \mu) \Sigma Q''(\zeta^* - \mu)^T)$$

• Using a similar approach to Capizzi and Masarotto (2016), a stopping criterion can be defined as

$$\bar{N}_s = \inf \left\{ k > N_m + N_f : k \geq \left( \frac{Z}{\nu} \right)^2 \max_{j=1,\ldots,p} \frac{1}{N-N_f} \sum_{\ell=N_f+1}^k \bar{g}_{j\ell}(\hat{\zeta}_k)^2 \right\},$$

• $z$ is the $[(1 - \nu)/2]$-th quantile of the standard normal distribution.

• $N_m + N_f$ is specified to avoid a premature ending of the algorithm.
Convergence criterion ii

“Deterministic” convergence criterion

- The “stochastic” convergence criterion $\bar{N}_s$ is coupled with a classical “deterministic” convergence criterion,

$$\bar{N}_a = \inf \left\{ k > N_m + N_f : \| \tilde{\zeta}_k - \tilde{\zeta}_{k-1} \| < \varepsilon \right\}.$$ 

Convergence criterion

- The convergence criterion used in our simulation is $\bar{N} = \min \left\{ \bar{N}_s, \bar{N}_a \right\}$.
- The convergence criteria $\bar{N}_s$ and $\bar{N}_a$ are “complementary”:
  - $\bar{N}_s$ is useful for optimizing “flat” functions, where the variance is small.
  - $\bar{N}_a$ can help when functions have high curvature, because $\tilde{\zeta}_k$ will “jump around” the optimum and the effect will be averaged out in $\tilde{\zeta}_k$ (Maryak, 1997).
“Deterministic” convergence criterion

- The “stochastic” convergence criterion $\tilde{N}_s$ is coupled with a classical “deterministic” convergence criterion,

$$\tilde{N}_a = \inf \left\{ k > N_m + N_f : \| \zeta_k - \zeta_{k-1} \| < \varepsilon \right\} .$$

Convergence criterion

- The convergence criterion used in our simulation is $\bar{N} = \min \{ \bar{N}_s, \bar{N}_a \}$.

- The convergence criteria $\bar{N}_s$ and $\bar{N}_a$ are “complementary”:
  - $\bar{N}_s$ is useful for optimizing “flat” functions, where the variance is small.
  - $\bar{N}_a$ can help when functions have high curvature, because $\tilde{\zeta}_k$ will “jump around” the optimum and the effect will be averaged out in $\tilde{\zeta}_k$ (Maryak, 1997).
“Deterministic” convergence criterion

• The “stochastic” convergence criterion $\bar{N}_s$ is coupled with a classical “deterministic” convergence criterion,

$$\bar{N}_a = \inf \left\{ k > N_m + N_t : \|\hat{\zeta}_k - \hat{\zeta}_{k-1}\| < \varepsilon \right\} .$$

Convergence criterion

• The convergence criterion used in our simulation is $\bar{N} = \min \{ \bar{N}_s, \bar{N}_a \}$.

• The convergence criteria $\bar{N}_s$ and $\bar{N}_a$ are “complementary”:
  • $\bar{N}_s$ is useful for optimizing “flat” functions, where the variance is small.
  • $\bar{N}_a$ can help when functions have high curvature, because $\hat{\zeta}_k$ will “jump around” the optimum and the effect will be averaged out in $\hat{\zeta}_k$ (Maryak, 1997).
“Deterministic” convergence criterion

- The “stochastic” convergence criterion $\bar{N}_s$ is coupled with a classical “deterministic” convergence criterion,

$$\bar{N}_a = \inf \left\{ k > N_m + N_f : \| \hat{\zeta}_k - \hat{\zeta}_{k-1} \| < \varepsilon \right\}.$$

Convergence criterion

- The convergence criterion used in our simulation is $\bar{N} = \min \left\{ \bar{N}_s, \bar{N}_a \right\}$.

- The convergence criteria $\bar{N}_s$ and $\bar{N}_a$ are “complementary”:
  - $\bar{N}_s$ is useful for optimizing “flat” functions, where the variance is small.
  - $\bar{N}_a$ can help when functions have high curvature, because $\hat{\zeta}_k$ will “jump around” the optimum and the effect will be averaged out in $\hat{\zeta}_k$ (Maryak, 1997).
“Deterministic” convergence criterion

• The “stochastic” convergence criterion $\bar{N}_s$ is coupled with a classical “deterministic” convergence criterion,

$$\bar{N}_a = \inf \left\{ k > N_m + N_f : \| \zeta_k - \zeta_{k-1} \| < \varepsilon \right\}.$$

Convergence criterion

• The convergence criterion used in our simulation is $\bar{N} = \min \left\{ \bar{N}_s, \bar{N}_a \right\}$.

• The convergence criteria $\bar{N}_s$ and $\bar{N}_a$ are “complementary”:
  • $\bar{N}_s$ is useful for optimizing “flat” functions, where the variance is small.
  • $\bar{N}_a$ can help when functions have high curvature, because $\hat{\zeta}_k$ will “jump around” the optimum and the effect will be averaged out in $\hat{\zeta}_k$ (Maryak, 1997).
Numerical results
Comparison with traditional methods

Grid search

- At iteration $k$, divide each coordinate of a grid $[\zeta_{\text{min}}^{(k)}, \zeta_{\text{max}}^{(k)}]$ in $m$ segments and calculate the objective function at the endpoints.
  - Find the endpoint $\zeta^{(k+1)}$ with minimum value of the objective.
  - The endpoints adjacent to $\zeta^{(k+1)}$ define $[\zeta_{\text{min}}^{(k+1)}, \zeta_{\text{max}}^{(k+1)}]$.

Nelder-Mead

- Method based on reflection, extension, contraction, and shrinkage of a simplex.
- An efficient implementation is available in the NLopt.jl package.
Comparison with traditional methods

Grid search

- At iteration $k$, divide each coordinate of a grid $[\zeta_{\text{min}}^{(k)}, \zeta_{\text{max}}^{(k)}]$ in $m$ segments and calculate the objective function at the endpoints.
- Find the endpoint $\zeta^{(k+1)}$ with minimum value of the objective.
- The endpoints adjacent to $\zeta^{(k+1)}$ define $[\zeta_{\text{min}}^{(k+1)}, \zeta_{\text{max}}^{(k+1)}]$.

Nelder-Mead

- Method based on reflection, extension, contraction, and shrinkage of a simplex.
- An efficient implementation is available in the NLopt.jl package.
Comparison with traditional methods

Grid search

- At iteration $k$, divide each coordinate of a grid $[\zeta_{min}^{(k)}, \zeta_{max}^{(k)}]$ in $m$ segments and calculate the objective function at the endpoints.
- Find the endpoint $\zeta^{(k+1)}$ with minimum value of the objective.
- The endpoints adjacent to $\zeta^{(k+1)}$ define $[\zeta_{min}^{(k+1)}, \zeta_{max}^{(k+1)}]$.

Nelder-Mead

- Method based on reflection, extension, contraction, and shrinkage of a simplex.
- An efficient implementation is available in the NLopt.jl package.
Comparison with traditional methods

Grid search

- At iteration $k$, divide each coordinate of a grid $[\zeta_{\text{min}}^{(k)}, \zeta_{\text{max}}^{(k)}]$ in $m$ segments and calculate the objective function at the endpoints.
- Find the endpoint $\zeta^{(k+1)}$ with minimum value of the objective.
- The endpoints adjacent to $\zeta^{(k+1)}$ define $[\zeta_{\text{min}}^{(k+1)}, \zeta_{\text{max}}^{(k+1)}]$.

Nelder-Mead

- Method based on reflection, extension, contraction, and shrinkage of a simplex.
  - An efficient implementation is available in the NLopt.jl package.
Comparison with traditional methods

Grid search

• At iteration $k$, divide each coordinate of a grid $[\zeta_{\min}^{(k)}, \zeta_{\max}^{(k)}]$ in $m$ segments and calculate the objective function at the endpoints.
• Find the endpoint $\zeta^{(k+1)}$ with minimum value of the objective.
• The endpoints adjacent to $\zeta^{(k+1)}$ define $[\zeta_{\min}^{(k+1)}, \zeta_{\max}^{(k+1)}]$.

Nelder-Mead

• Method based on reflection, extension, contraction, and shrinkage of a simplex.
• An efficient implementation is available in the NLopt.jl package.
• CUSUM control chart: \( C_t = \max \{0, C_{t-1} + X_t - k\} \), \( X_t \sim N(\delta, 1) \) for various \( \delta \)'s.
• Optimal parameter to detect the mean shift \( \delta \) is \( k^* = \delta / 2 \).

**Figure 1:** Estimated optimal parameter values over 100 optimizations.
Figure 2: Computing times over 100 optimization.
Multidimensional scalability

- MEWMA control chart: $C_t = (I - \Lambda)C_{t-1} + \Lambda X_t$, with $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_d)$ and $\lambda_j \in (0, 1)$ for all $j$. $X_{t,1} \sim \mathcal{N}(\delta, 1), X_{t,2} \sim \chi^2_{1+\delta \sqrt{2}}, X_{t,3} \sim \text{Pois}(1 + \delta)$.

**Figure 3:** Median, 0.1th and 0.9th quantiles of the computing times over 100 optimizations.
Extension to other metrics

Generalizations

• Sometimes, other metrics such as the median run length or quantiles of the run length are of interest (Knoth, 2015).

• The proposed SPSA algorithm is flexible enough to be generalized to other performance metrics based on the RL.

Example: median run length

• Minimization of the out-of-control median run length with a constraint on the in-control median run length.

• $Q$ becomes the median of the $r$ simulated out-of-control run lengths.

• The control limit is estimated using a modified gradient iteration in the SA algorithm (Capizzi and Masarotto, 2009)
Extension to other metrics

Generalizations

- Sometimes, other metrics such as the median run length or quantiles of the run length are of interest (Knoth, 2015).
- The proposed SPSA algorithm is flexible enough to be generalized to other performance metrics based on the RL.

Example: median run length

- Minimization of the out-of-control median run length with a constraint on the in-control median run length.
- \( Q \) becomes the median of the \( r \) simulated out-of-control run lengths.
- The control limit is estimated using a modified gradient iteration in the SA algorithm (Capizzi and Masarotto, 2009)
Extension to other metrics

Generalizations

• Sometimes, other metrics such as the median run length or quantiles of the run length are of interest (Knoth, 2015).

• The proposed SPSA algorithm is flexible enough to be generalized to other performance metrics based on the RL.

Example: median run length

• Minimization of the out-of-control median run length with a constraint on the in-control median run length.

• \( Q \) becomes the median of the \( r \) simulated out-of-control run lengths.

• The control limit is estimated using a modified gradient iteration in the SA algorithm (Capizzi and Masarotto, 2009)
Extension to other metrics

Generalizations

- Sometimes, other metrics such as the median run length or quantiles of the run length are of interest (Knoth, 2015).
- The proposed SPSA algorithm is flexible enough to be generalized to other performance metrics based on the RL.

Example: median run length

- Minimization of the out-of-control median run length with a constraint on the in-control median run length.
- \( Q \) becomes the median of the \( r \) simulated out-of-control run lengths.
- The control limit is estimated using a modified gradient iteration in the SA algorithm (Capizzi and Masarotto, 2009).
Extension to other metrics

Generalizations

- Sometimes, other metrics such as the median run length or quantiles of the run length are of interest (Knoth, 2015).

- The proposed SPSA algorithm is flexible enough to be generalized to other performance metrics based on the RL.

Example: median run length

- Minimization of the out-of-control median run length with a constraint on the in-control median run length.

- $Q$ becomes the median of the $r$ simulated out-of-control run lengths.

- The control limit is estimated using a modified gradient iteration in the SA algorithm (Capizzi and Masarotto, 2009)
Conclusion
Summary

• We have proposed a novel methodology for a more **efficient design** of control charts tuning parameters.

• The only requirement is to be able to **simulate run lengths** from the IC process (e.g. using parametric/nonparametric bootstrap, bootstrap for time series, ...)

• The methodology is based on a **stochastic approximations** algorithm that is specifically designed for the constrained optimization problem.

• Numerical simulation suggest that the approach is **more efficient** than traditional approaches, especially for **multi-dimensional** tuning parameters.

• Due to its flexibility, it can be **generalized** to the optimization of various performance metrics such as median run length, run length quantiles, etc.
Summary

• We have proposed a novel methodology for a more efficient design of control charts tuning parameters.

• The only requirement is to be able to simulate run lengths from the IC process (e.g. using parametric/nonparametric bootstrap, bootstrap for time series, ...)

• The methodology is based on a stochastic approximations algorithm that is specifically designed for the constrained optimization problem.

• Numerical simulation suggest that the approach is more efficient than traditional approaches, especially for multi-dimensional tuning parameters.

• Due to its flexibility, it can be generalized to the optimization of various performance metrics such as median run length, run length quantiles, etc.
Summary

• We have proposed a novel methodology for a more **efficient design** of control charts tuning parameters.

• The only requirement is to be able to **simulate run lengths** from the IC process (e.g. using parametric/nonparametric bootstrap, bootstrap for time series, …)

• The methodology is based on a **stochastic approximations** algorithm that is specifically designed for the constrained optimization problem.

• Numerical simulation suggest that the approach is **more efficient** than traditional approaches, especially for **multi-dimensional** tuning parameters.

• Due to its flexibility, it can be **generalized** to the optimization of various performance metrics such as median run length, run length quantiles, etc.
Summary

• We have proposed a novel methodology for a more **efficient design** of control charts tuning parameters.

• The only requirement is to be able to **simulate run lengths** from the IC process (e.g. using parametric/nonparametric bootstrap, bootstrap for time series, …)

• The methodology is based on a **stochastic approximations** algorithm that is specifically designed for the constrained optimization problem.

• Numerical simulation suggest that the approach is **more efficient** than traditional approaches, especially for **multi-dimensional** tuning parameters.

• Due to its flexibility, it can be **generalized** to the optimization of various performance metrics such as median run length, run length quantiles, etc.
Summary

- We have proposed a novel methodology for a more efficient design of control charts tuning parameters.
- The only requirement is to be able to simulate run lengths from the IC process (e.g. using parametric/nonparametric bootstrap, bootstrap for time series, ...)
- The methodology is based on a stochastic approximations algorithm that is specifically designed for the constrained optimization problem.
- Numerical simulation suggest that the approach is more efficient than traditional approaches, especially for multi-dimensional tuning parameters.
- Due to its flexibility, it can be generalized to the optimization of various performance metrics such as median run length, run length quantiles, etc.
Thank you for the attention
Questions


The proposed SPSA algorithm requires the selection of many tuning constants.

A semi-automated way of selecting most of the tuning constants is available following the guidelines by Spall (1992), Spall (1998) and Spall (2003).

- The gain sequences in the algorithm are defined as $a_k = a/(k + A + 1)^\alpha$ and $c_k = c/(k + 1)^\beta$, where $\alpha$ and $\beta$ are pre-specified to be 0.602 and 0.101, respectively (Spall, 2003).
- These gain sequences would result in a slow gain decay and ensure the convergence of $\hat{\zeta}_k$ to $\zeta^*$ as $k \to \infty$ under some quite general assumptions, as proved by Spall (1992).
- The constant $a$, $A$ and $c$ require a small preliminary adaptive step in order to be estimated.
- $c$ can be approximately set to be the standard deviation $\sigma_{\hat{\zeta}_0}$ of the OC RL calculated at the initial value $\hat{\zeta}_0$ (Spall, 1998).
- In numerical studies, we have seen that setting $c = \min\{\sigma_{\hat{\zeta}_0}, 0.1\}$ can avoid excessive perturbation of the tuning parameters in the early iterations.
- $A$ can be set to be 0.1 times the expected number of function evaluations. For example, the expected number of evaluations used in this paper is 150, resulting in $A = 0.1 \times 150 = 15$. 
(Spall, 1998) recommends selecting $a$ to be the expected magnitude change in $\hat{\zeta}_k$ during the first few iterations. Specifically,

$$a = s \cdot (A + 1)^\alpha / \bar{G},$$

where $s$ is the initial step size and $\bar{G} = \frac{1}{d} \sum_{j=1}^{d} \sum_{i=1}^{n_c} g_{ij}(\hat{\zeta}_0)/n_c$ is a preliminary estimate of the average value of the gradient in $\hat{\zeta}_0$ based on $n_c$ simulated RLs.

For instance, a reasonable initial step size $s$ for an EWMA chart could be 0.2, and setting $n_c = 20$ is found to be appropriate to estimate the gradient at the beginning of the algorithm.