Simulation laboratory 4: Variance reduction

Yuki Oyama

Transport and Mobility Laboratory
School of Architecture, Civil and Environmental Engineering
École Polytechnique Fédérale de Lausanne

March 6, 2018

Goals

Variance reduction:

- Understand two different variance reduction techniques
- Increase the precision of a vehicle queue simulation result

Implementation:

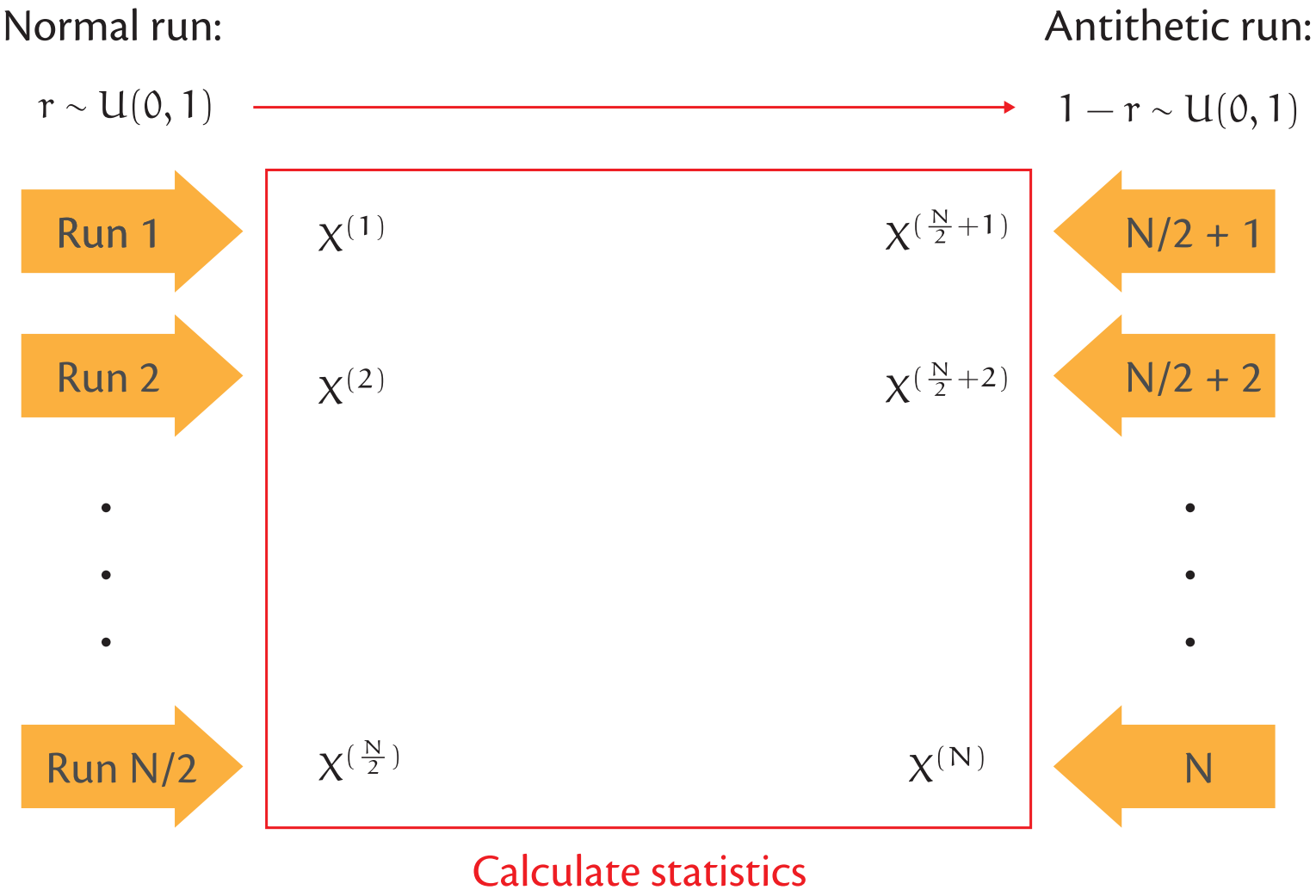
1. Antithetic draws
2. Control variates

1 Antithetic draws

2 Control variates

3 My results

Antithetic draws



Antithetic draws

Implementation:

- Extend the function `QueuingSimulation1()` to:
  1. **QueuingSimulation1_Independent.m**: as a function with
     - Input: `scenario`
     - Output: `[times, queues, rands]`
  2. **QueuingSimulation1_Antithetic.m**: as a function with
     - Input: `scenario, rands`
     - Output: `[times, queues]`
- Test file is also an extension of
  - **StatAnalysisAndBootstrapTest.m**

Independent draws workflow

1. For $r = 1, \ldots, N$:
   1. Run the simulation `QueueingSimulation1(scenario)`
   2. Obtain max-queue-length $q_{\max}^{(r)}$
2. Calculate the statistics of $[q_{\max}^{(1)}, \ldots, q_{\max}^{(N)}]$

- N is defined by the stopping criteria.

Antithetic draws workflow

1. For $r = 1, \ldots, \mathbf{N/2}$:
   1. Run `QueueingSimulation1_Independent(scenario)`
      - **Keep the draws** used in the simulation $u_{e,i}^{(r)} \sim U(0,1)$, $e \in \{\text{Generation}, \text{Arrival}, \text{Departure}\}$, $i = 1, \ldots, I_r$.
   2. Obtain max-queue-length $q_{\max}^{(r)}$
2. For $r = \mathbf{(N/2 + 1)}, \ldots, N$:
   1. Run `QueueingSimulation1_Antithetic(scenario,` $u^{(r-N/2)}$`)`.
      - $1 - u^{(r-N/2)}$ is used for the events simulation
      - If $I_r > I_{r-N/2}$, draw new random numbers for $i = (I_{r-N/2} + 1), \ldots, I_r$.
   2. Obtain max-queue-length $q_{\max}^{(r)}$
3. Calculate the statistics of $[q_{\max}^{(1)}, \ldots, q_{\max}^{(N)}]$

Keep draws

- The negative correlation between a run and its antithetic should be as big as possible.
- The order of events could change between the two runs.
- **Different streams of random numbers for the different event types should be recorded**.
- `QueuingSimulation_Independent()` outputs the vectors of draws:
  - `rands.Generationrands`
  - `rands.Arrivalrands`
  - `rands.Departurerands`

1 Antithetic draws

2 Control variates

3 My results

Control variates

Run 1 $\quad X^{(1)} \longrightarrow$

Run 2 $\quad X^{(2)} \longrightarrow$

$\vdots$

Run N $\quad X^{(N)} \longrightarrow$

Control variate draw

$$Z^{(1)} = X^{(1)} + c^{(1)}(Y^{(1)} - \mu)$$

Control variate expectation

$$Z^{(2)} = X^{(2)} + c^{(2)}(Y^{(2)} - \mu)$$

$$Z^{(N)} = X^{(N)} + c^{(N)}(Y^{(N)} - \mu)$$

Calculate statistics

Control variates

Implementation:

- Extend the function `QueuingSimulation1()` to:
  - **QueuingSimulation1_ControlVariates.m**: as a function with
    - Input: `scenario`
    - Output: `[times, queues, control (Y)]`
- Implement a function to calculate $Z = X + c(Y - \mu)$:
  - **ControlledMean.m**
- Test file is also an extension of
  - **StatAnalysisAndBootstrapTest.m**

Control variates workflow

1. For $r = 1, \ldots, N$:
   1. Run `QueueingSimulation1_ControlVariates(scenario)`
      - Output control variate `control` $(= Y)$
   2. Obtain max-queue-length $q_{\max}$ $(= X)$
   3. Run `ControlledMean(X, Y, μ)`, in which
      1. Calculate variance $\mathrm{Var}(X), \mathrm{Var}(Y)$ and covariance $\mathrm{Cov}(X, Y)$
      2. Define the constant $c^* = \mathrm{Cov}(X, Y)/\mathrm{Var}(Y)$
      3. Calculate $Z = X + c^*(Y - \mu)$
      4. Obtain the average and variance of $[Z^{(1)}, \ldots, Z^{(r)}]$

Control variates

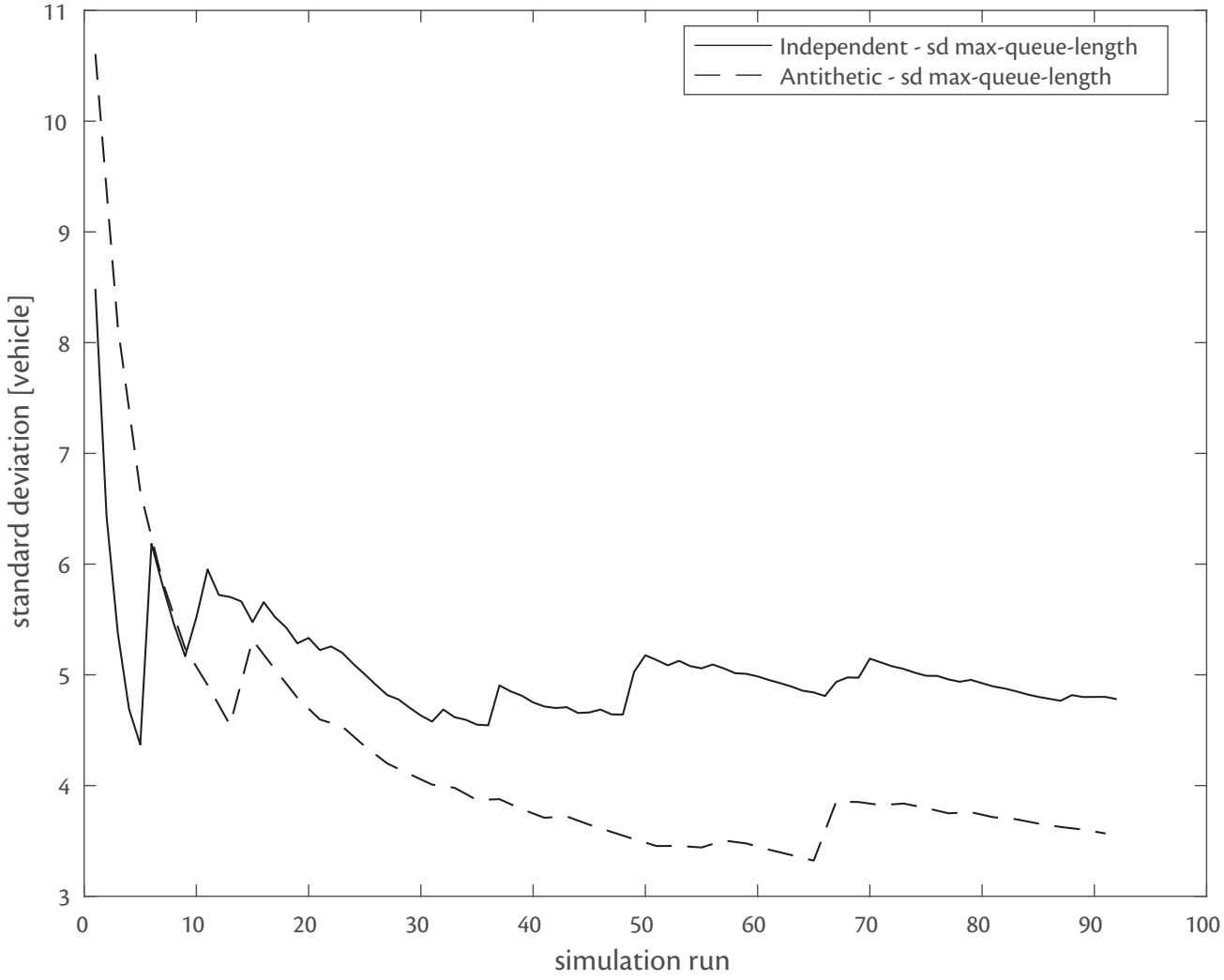
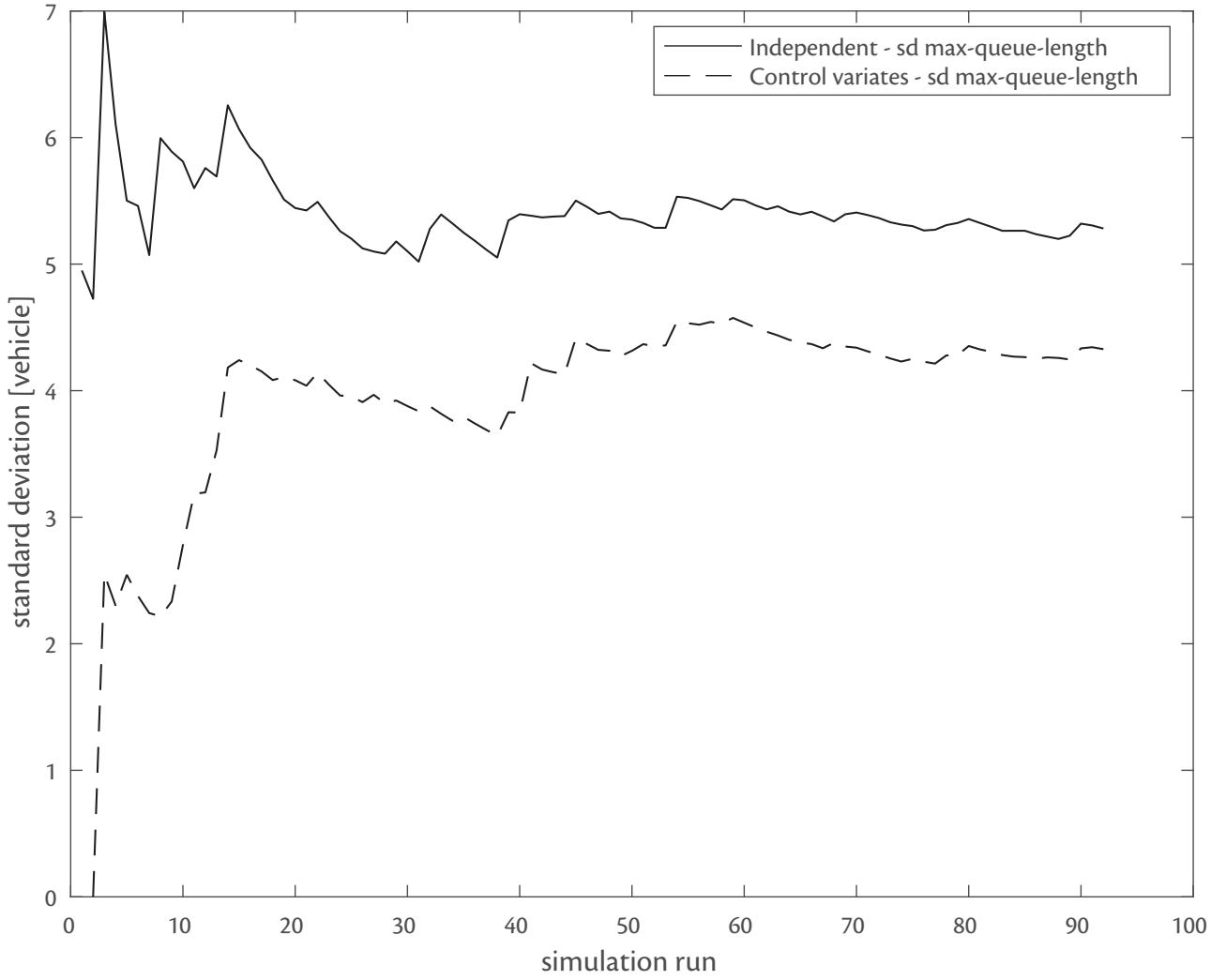
- Choose the control variate (Y) as you prefer. The higher the correlation is, the bigger the variance reduction is.
- Example: $Y =$ mean of service time at bottleneck

1 Antithetic draws

2 Control variates

3 My results

Antithetic draws



Control variates