Modeling and Optimization Problems in Contact Centers (a biased overview)

Pierre L'Ecuyer

Canada Research Chair in Stochastic Simulation and Optimization, U. Montréal

Sponsored by Bell Canada

Modeling and Optimization Problems in Contact Centers (a biased overview)

Pierre L'Ecuyer

Canada Research Chair in Stochastic Simulation and Optimization, U. Montréal

Sponsored by Bell Canada

- General overview and problem statements; importance of contact centers;
- Quantitative evaluation: queueing approximations vs simulation;
- Building realistic models;
- Optimization of staffing, scheduling, call routing, priorities, outbound call, etc.
- Simulation tools and improving simulation efficiency;

For my articles, Google "Pierre L'Ecuyer"

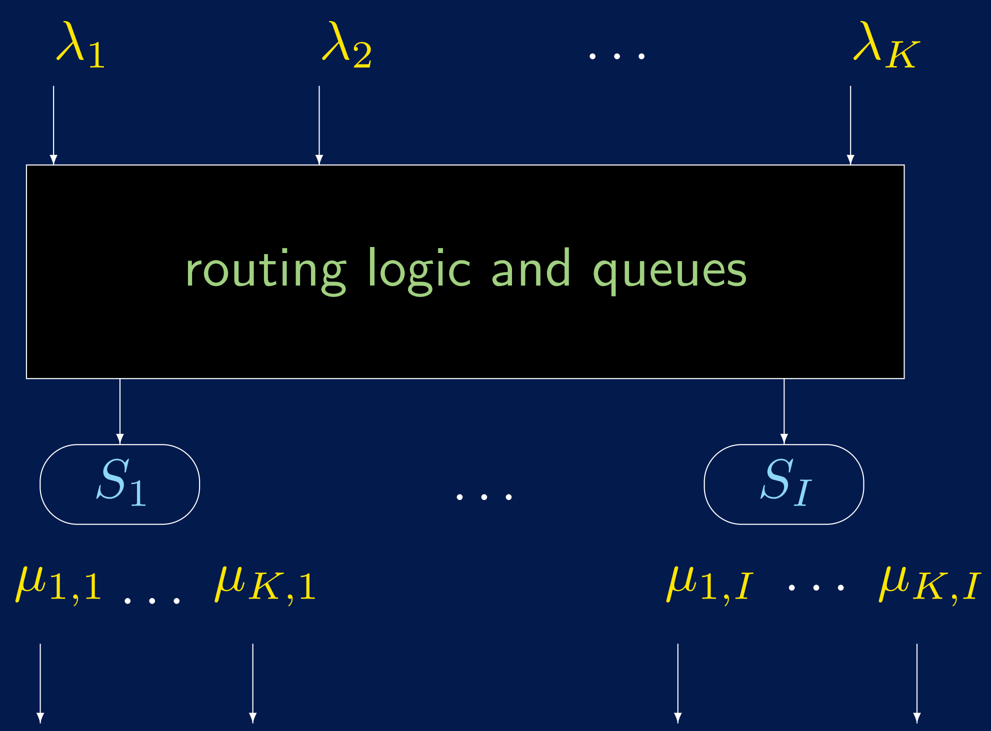
See also the references in the paper: Gans, Koole, Mandelbaum, Whitt, etc.

Example: A Call Center with Multiple Call Types

K call types. Depends on required technical skill, language, importance, etc.
I agent types (or skill groups). Each has skills to handle certain call types.
Some agents may be better than others for a given call type.

Example: A Call Center with Multiple Call Types

K call types. Depends on required technical skill, language, importance, etc.
I agent types (or skill groups). Each has skills to handle certain call types.
Some agents may be better than others for a given call type.



Call arrivals: follow some stochastic process.
Arrival rates change with time and are also stochastic (high uncertainty)!

Call arrivals: follow some stochastic process.
Arrival rates change with time and are also stochastic (high uncertainty)!

Abandonments: callers may abandon after a random patience time.

Retrials and returns: call again (often neglected...).

Call arrivals: follow some stochastic process.
Arrival rates change with time and are also stochastic (high uncertainty)!

Abandonments: callers may abandon after a random patience time.

Retrials and returns: call again (often neglected...).

If each agent has a single skill: K single queues in parallel.

Call arrivals: follow some stochastic process.
Arrival rates change with time and are also stochastic (high uncertainty)!

Abandonments: callers may abandon after a random patience time.

Retrials and returns: call again (often neglected...).

If each agent has a single skill: K single queues in parallel.

If each agent has all skills: one single queue.
More efficient (less wait, fewer abandonments), but more costly.

Each additional skill increases the cost of an agent.
Can also decrease its speed!

Call arrivals: follow some stochastic process.
Arrival rates change with time and are also stochastic (high uncertainty)!

Abandonments: callers may abandon after a random patience time.

Retrials and returns: call again (often neglected...).

If each agent has a single skill: K single queues in parallel.

If each agent has all skills: one single queue.
More efficient (less wait, fewer abandonments), but more costly.

Each additional skill increases the cost of an agent.
Can also decrease its speed!

For well-balanced systems, one or two skills per agent often gives a performance almost as good as all skills for all agents (e.g., Wallace and Whitt 2004).

Performance measures

Example: Total cost of agents.

Performance measures

Example: Total cost of agents.

Constraints

Service level (SL): fraction of calls that wait less than acceptable waiting time τ (typically 20 to 30 seconds). Often measured and controlled separately by time period (hour, day, ...), by call type, ... $+$ aggregated. (Government regulations.)

Performance measures

Example: Total cost of agents.

Constraints

Service level (SL): fraction of calls that wait less than acceptable waiting time τ (typically 20 to 30 seconds). Often measured and controlled separately by time period (hour, day, ...), by call type, ... $+$ aggregated. (Government regulations.)

Caveat: could cheat and never serve calls that already waited too much.

Performance measures

Example: Total cost of agents.

Constraints

Service level (SL): fraction of calls that wait less than acceptable waiting time τ (typically 20 to 30 seconds). Often measured and controlled separately by time period (hour, day, ...), by call type, ... + aggregated. (Government regulations.)

Caveat: could cheat and never serve calls that already waited too much.

Expected excess waiting time: $\mathbb{E}[\max(0, \text{Wait} - \tau)]$.

Performance measures

Example: Total cost of agents.

Constraints

Service level (SL): fraction of calls that wait less than acceptable waiting time τ (typically 20 to 30 seconds). Often measured and controlled separately by time period (hour, day, ...), by call type, ... + aggregated. (Government regulations.)

Caveat: could cheat and never serve calls that already waited too much.

Expected excess waiting time: $\mathbb{E}[\max(0, \text{Wait} - \tau)]$.

Abandonment ratio: fraction of calls that abandon.

Performance measures

Example: Total cost of agents.

Constraints

Service level (SL): fraction of calls that wait less than acceptable waiting time τ (typically 20 to 30 seconds). Often measured and controlled separately by time period (hour, day, ...), by call type, ... + aggregated. (Government regulations.)

Caveat: could cheat and never serve calls that already waited too much.

Expected excess waiting time: $\mathbb{E}[\max(0, \text{Wait} - \tau)]$.

Abandonment ratio: fraction of calls that abandon.

Occupation ratio of agents, per type and per period.
Should not exceed 90–92% (for fairness, stress, ...).

Skill-based routing (SBR) strategies:

Rules that control in real time the agent-to-call and call-to-agent assignments.
Act as controlling device for service levels, across call types and globally.

Skill-based routing (SBR) strategies:

Rules that control in real time the agent-to-call and call-to-agent assignments.
Act as controlling device for service levels, across call types and globally.

Dynamic routing: Decision may depend on the entire state of the system.
An optimal policy is generally too complicated and hard to implement.

Skill-based routing (SBR) strategies:

Rules that control in real time the agent-to-call and call-to-agent assignments.
Act as controlling device for service levels, across call types and globally.

Dynamic routing: Decision may depend on the entire state of the system.
An optimal policy is generally too complicated and hard to implement.

Static routing: Each call type has an ordered list of agent types.
If all are busy, the call joins a queue (usually one queue per call type).

Each agent type may also have an ordered list of queues (call types) for when it becomes available (priorities).

Skill-based routing (SBR) strategies:

Rules that control in real time the agent-to-call and call-to-agent assignments.
Act as controlling device for service levels, across call types and globally.

Dynamic routing: Decision may depend on the entire state of the system.
An optimal policy is generally too complicated and hard to implement.

Static routing: Each call type has an ordered list of agent types.
If all are busy, the call joins a queue (usually one queue per call type).

Each agent type may also have an ordered list of queues (call types) for when it becomes available (priorities).

Such hard priorities may easily lead to unstable systems.
Better to also look at queue lengths.

Skill-based routing (SBR) strategies:

Rules that control in real time the agent-to-call and call-to-agent assignments.
Act as controlling device for service levels, across call types and globally.

Dynamic routing: Decision may depend on the entire state of the system.
An optimal policy is generally too complicated and hard to implement.

Static routing: Each call type has an ordered list of agent types.
If all are busy, the call joins a queue (usually one queue per call type).

Each agent type may also have an ordered list of queues (call types) for when it becomes available (priorities).

Such hard priorities may easily lead to unstable systems.
Better to also look at queue lengths.

Example: Weight the calls (e.g., waiting time so far, multiplied by an importance factor that depends on call type).
Next available agent picks call with largest weight among the types he can handle.

Staffing problem: Divide the day into periods (e.g, half hours) and determine how many agents of each type to have in the center for each period in order to meet the performance constraints, at minimal cost.

Staffing problem: Divide the day into periods (e.g, half hours) and determine how many agents of each type to have in the center for each period in order to meet the performance constraints, at minimal cost.

The SL in one period can depend on the number of agents in other periods!

Staffing problem: Divide the day into periods (e.g, half hours) and determine how many agents of each type to have in the center for each period in order to meet the performance constraints, at minimal cost.

The SL in one period can depend on the number of agents in other periods!

Caveat: not possible to match exactly the optimal staffing by scheduling a set of agents whose working shifts are admissible (i.e., satisfy the constraints determined by Union rules, etc.).

Staffing problem: Divide the day into periods (e.g, half hours) and determine how many agents of each type to have in the center for each period in order to meet the performance constraints, at minimal cost.

The SL in one period can depend on the number of agents in other periods!

Caveat: not possible to match exactly the optimal staffing by scheduling a set of agents whose working shifts are admissible (i.e., satisfy the constraints determined by Union rules, etc.).

Scheduling problem: Determine a set of agents, each with its working shift for the day, so that the performance constraints are met, at minimal cost.
Warning: constraints on shifts should not be too restrictive!

Staffing problem: Divide the day into periods (e.g, half hours) and determine how many agents of each type to have in the center for each period in order to meet the performance constraints, at minimal cost.

The SL in one period can depend on the number of agents in other periods!

Caveat: not possible to match exactly the optimal staffing by scheduling a set of agents whose working shifts are admissible (i.e., satisfy the constraints determined by Union rules, etc.).

Scheduling problem: Determine a set of agents, each with its working shift for the day, so that the performance constraints are met, at minimal cost.
Warning: constraints on shifts should not be too restrictive!

Scheduling and rostering problem:
In practice, we do not have an infinite supply of each agent type!
For a given set of agents and a given set of admissible shifts, assign a shift to each agent, to meet the performance constraints at minimal cost.

Recourses against uncertainty

Arrival times are random and call volumes also vary unpredictably.
Some agents may fail to show up. Etc.

Recourses against uncertainty

Arrival times are random and call volumes also vary unpredictably.
Some agents may fail to show up. Etc.

Sharing resources provides economy of scale:
— merging two or more centers into a large one;
— agents with multiple skills.

Recourses against uncertainty

Arrival times are random and call volumes also vary unpredictably.
Some agents may fail to show up. Etc.

Sharing resources provides economy of scale:
— merging two or more centers into a large one;
— agents with multiple skills.

But what if the total volume of calls is much different than expected?
This happens frequently!

Recourses against uncertainty

Arrival times are random and call volumes also vary unpredictably.
Some agents may fail to show up. Etc.

Sharing resources provides economy of scale:
— merging two or more centers into a large one;
— agents with multiple skills.

But what if the total volume of calls is much different than expected?
This happens frequently!

Would need flexibility to change staffing levels on short notice.

Possibilities:
— move meetings and training sessions to low-volume periods;
— agents on standby at home;
— outsourcing part of the load (pass some of the risk, like with an insurance).

Blend systems: inbound and outbound calls

Can be used to "equalize" the workload w.r.t. time.

When inbound traffic is low, can handle faxes, emails, and/or make outbound call, trying to reach customers (e.g., for telemarketing, or to return calls).

Blend systems: inbound and outbound calls

Can be used to "equalize" the workload w.r.t. time.
When inbound traffic is low, can handle faxes, emails, and/or make outbound call, trying to reach customers (e.g., for telemarketing, or to return calls).

A predictive dialer makes bunches of outbound call attempts.

Blend systems: inbound and outbound calls

Can be used to "equalize" the workload w.r.t. time.
When inbound traffic is low, can handle faxes, emails, and/or make outbound call, trying to reach customers (e.g., for telemarketing, or to return calls).

A predictive dialer makes bunches of outbound call attempts.

A right party connect: when the outbound contact is successful (good).

Blend systems: inbound and outbound calls

Can be used to "equalize" the workload w.r.t. time.
When inbound traffic is low, can handle faxes, emails, and/or make outbound call, trying to reach customers (e.g., for telemarketing, or to return calls).

A predictive dialer makes bunches of outbound call attempts.

A right party connect: when the outbound contact is successful (good).

A mismatch: when the successful contact cannot be served immediately (bad).

Blend systems: inbound and outbound calls

Can be used to "equalize" the workload w.r.t. time.
When inbound traffic is low, can handle faxes, emails, and/or make outbound call, trying to reach customers (e.g., for telemarketing, or to return calls).

A predictive dialer makes bunches of outbound call attempts.

A right party connect: when the outbound contact is successful (good).

A mismatch: when the successful contact cannot be served immediately (bad).

Possible **additional constraints**:
– Lower bound on the (expected) volume of outbound calls per day.
– Upper bound on the (expected) fraction of mismatches.

Blend systems: inbound and outbound calls

Can be used to "equalize" the workload w.r.t. time.
When inbound traffic is low, can handle faxes, emails, and/or make outbound call, trying to reach customers (e.g., for telemarketing, or to return calls).

A predictive dialer makes bunches of outbound call attempts.

A right party connect: when the outbound contact is successful (good).

A mismatch: when the successful contact cannot be served immediately (bad).

Possible **additional constraints**:
– Lower bound on the (expected) volume of outbound calls per day.
– Upper bound on the (expected) fraction of mismatches.

Other types of recourses:

Controlling the arrival rate, e.g., by giving information on current waiting times?

Tell people to call again later.

Other ways?

Besides routing and scheduling:

Long-term strategic decisions

- Size and layout of center
- Call types, skill groups
- Types of work schedules
- Outsourcing decisions and contracts

Medium-term planning

- Hiring and training of new agents
- Training current agents

Building realistic models

We need detailed data about the call center operations:
arrivals, service times, breaks, hidden rules and practice, etc.

Building realistic models

We need detailed data about the call center operations:
arrivals, service times, breaks, hidden rules and practice, etc.

With partial and aggregated information, we can only build an "approximate" model, which may be unrealistic.

This is often a problem!

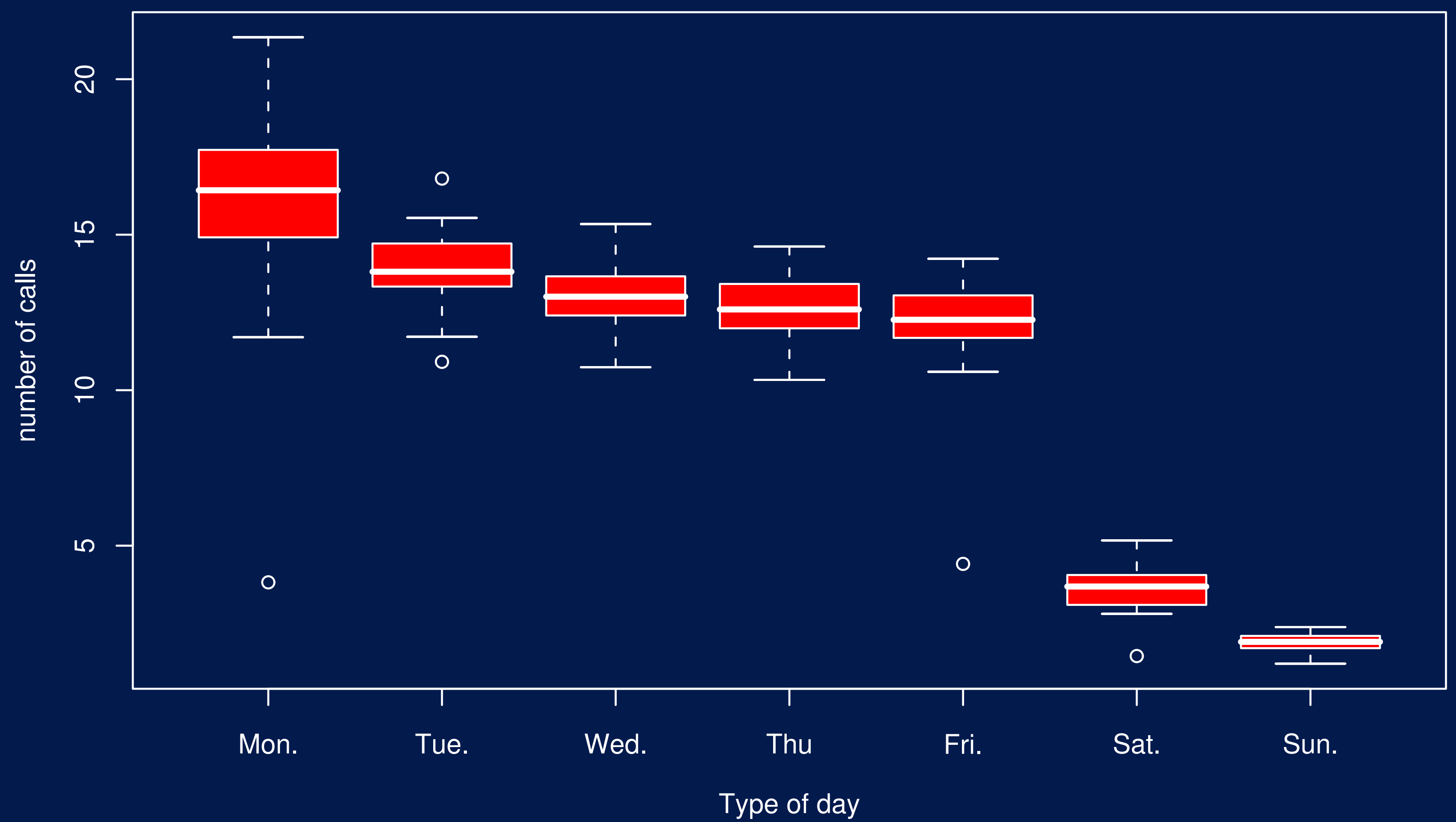
Building realistic models

We need detailed data about the call center operations:
arrivals, service times, breaks, hidden rules and practice, etc.

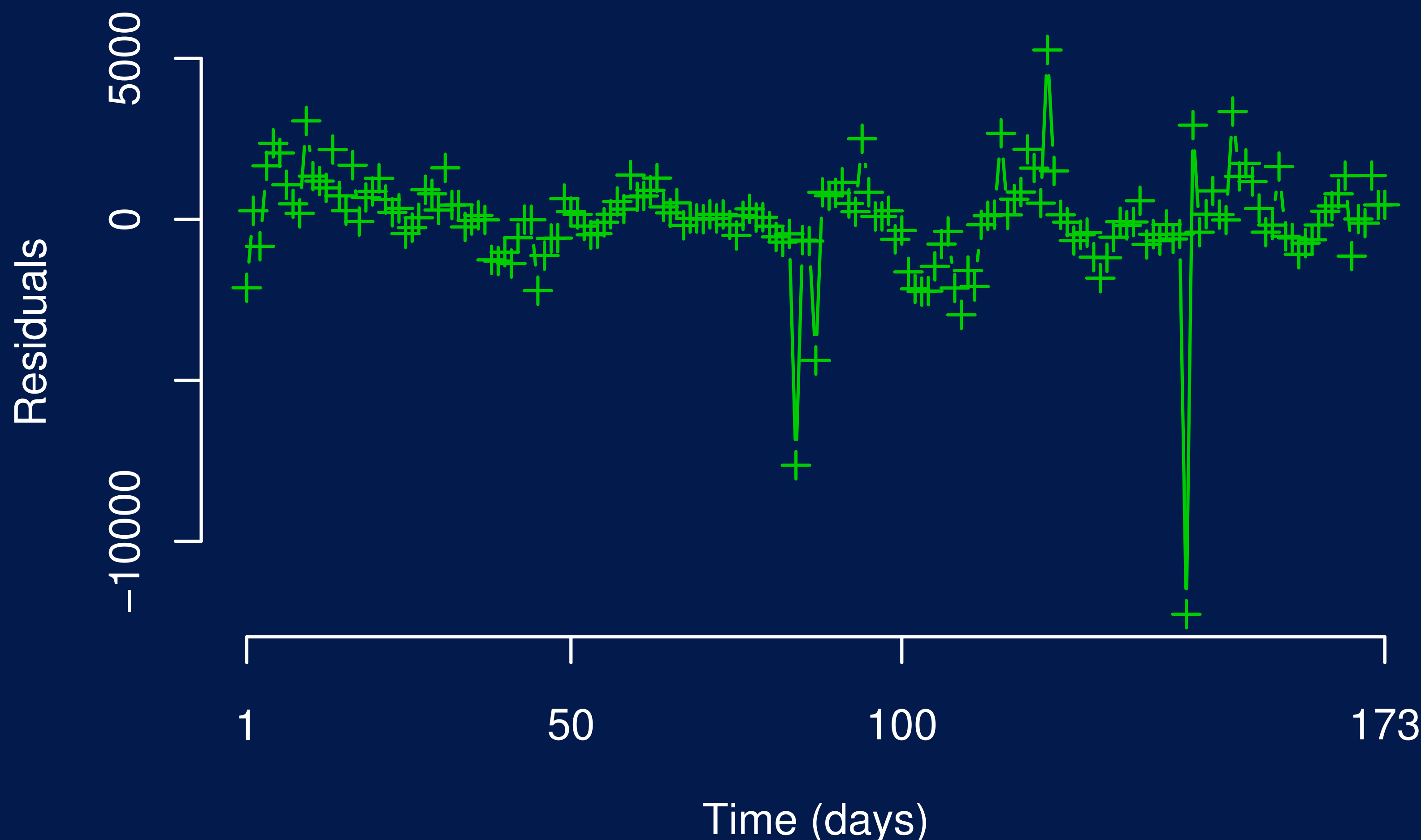
With partial and aggregated information, we can only build an "approximate" model, which may be unrealistic.

This is often a problem!

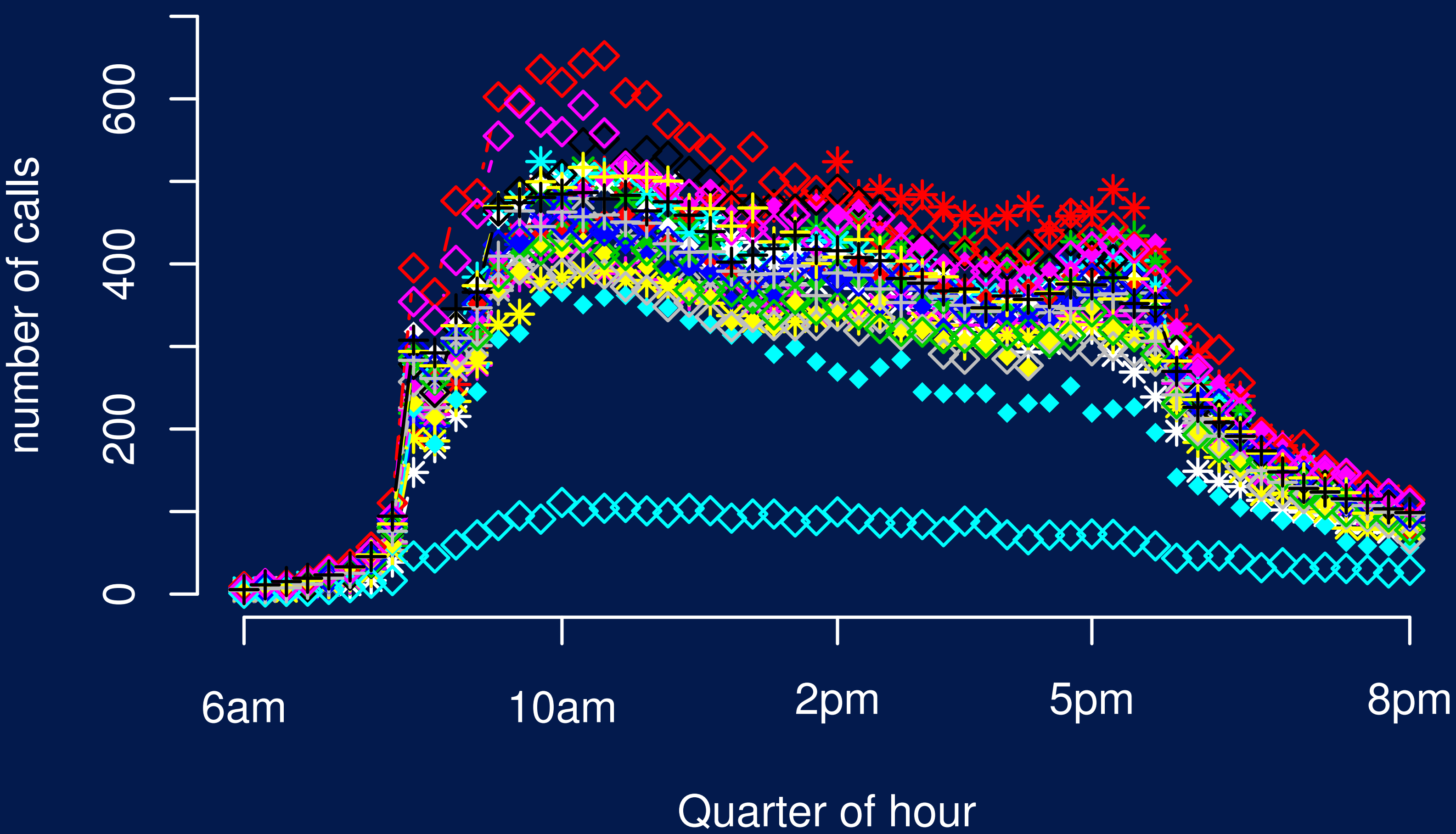
Reality: service times are not exponential; arrivals are not Poisson and stationary, breaks are not exactly as planned, agents are not available exactly as planned (e.g., they may sometimes disconnect themselves for a short period between two calls), etc.



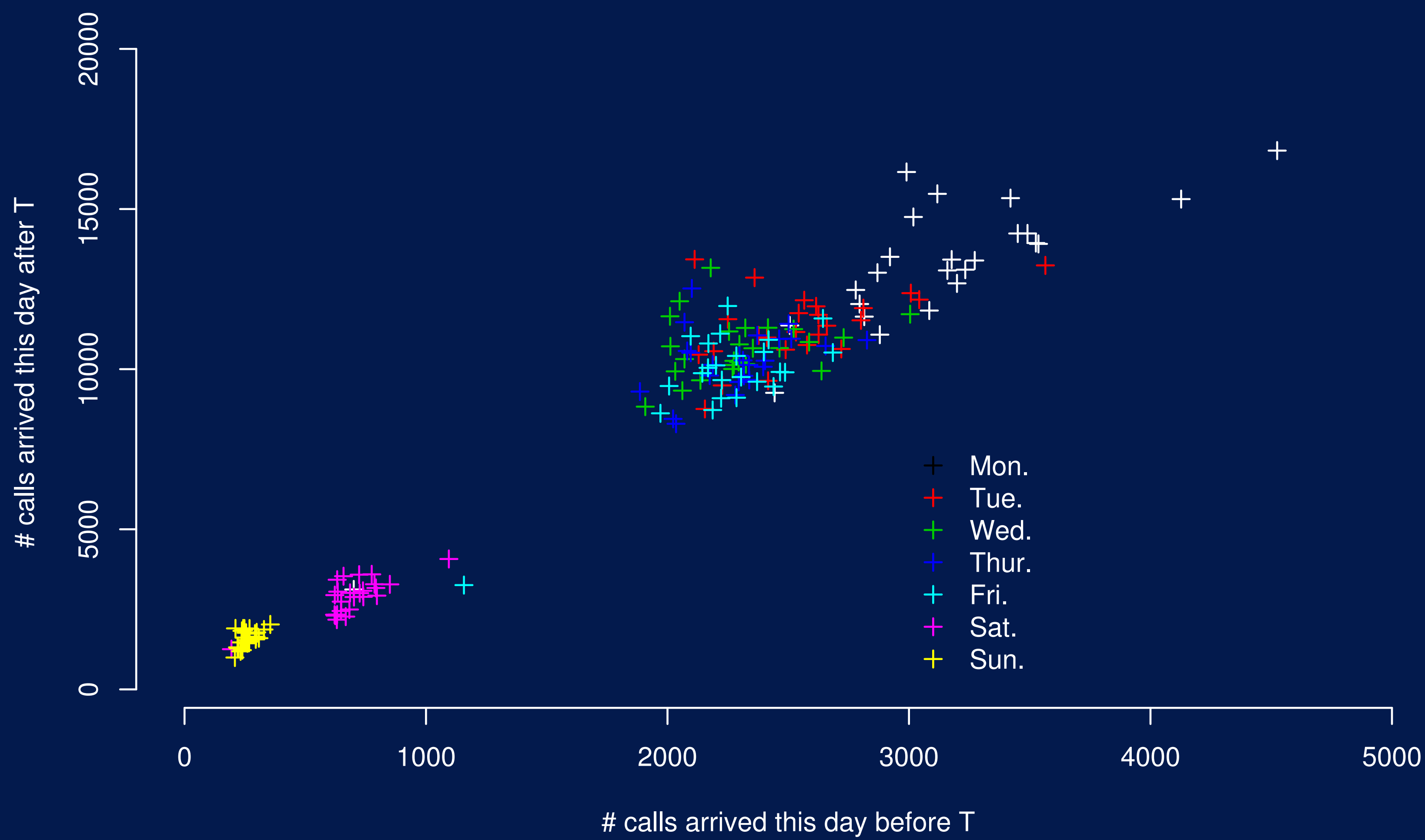
Arrival counts on different days



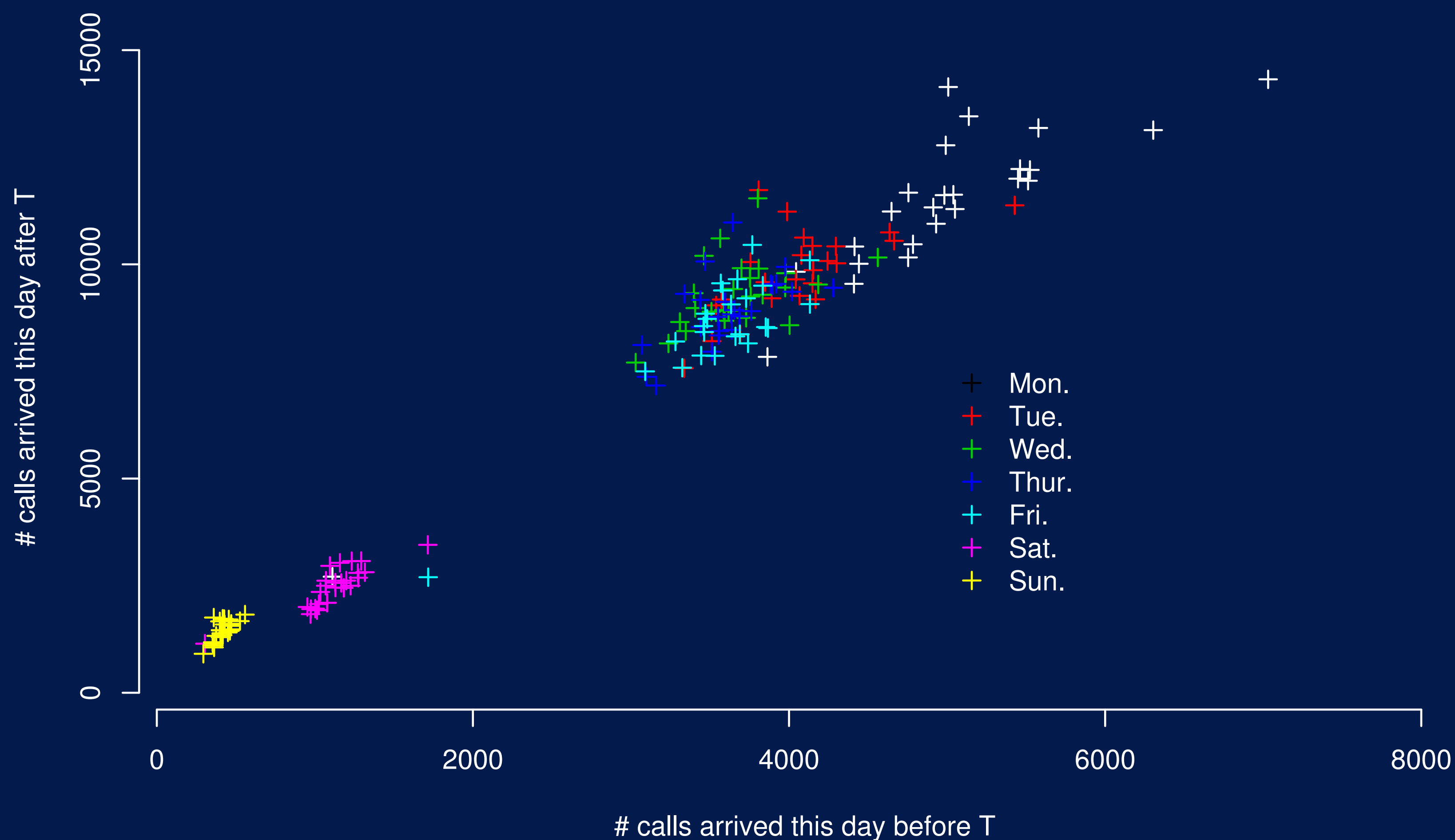
Correlations between successive days:
Daily residuals after removing day-effect, Box-Ljung test: p-value $= 7.5 \times 10^{-6}$



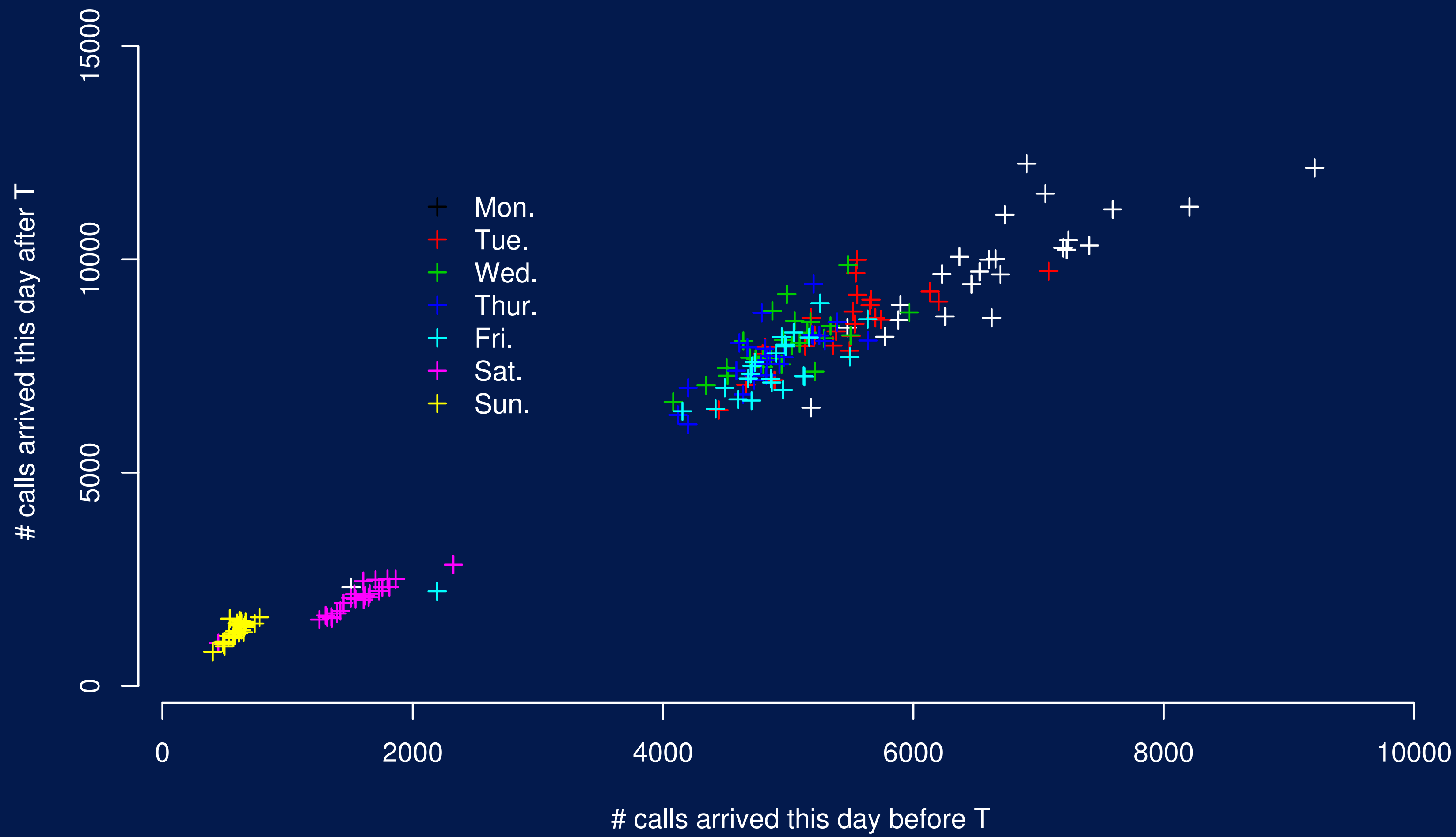
Monday, arrival counts by quarter hour



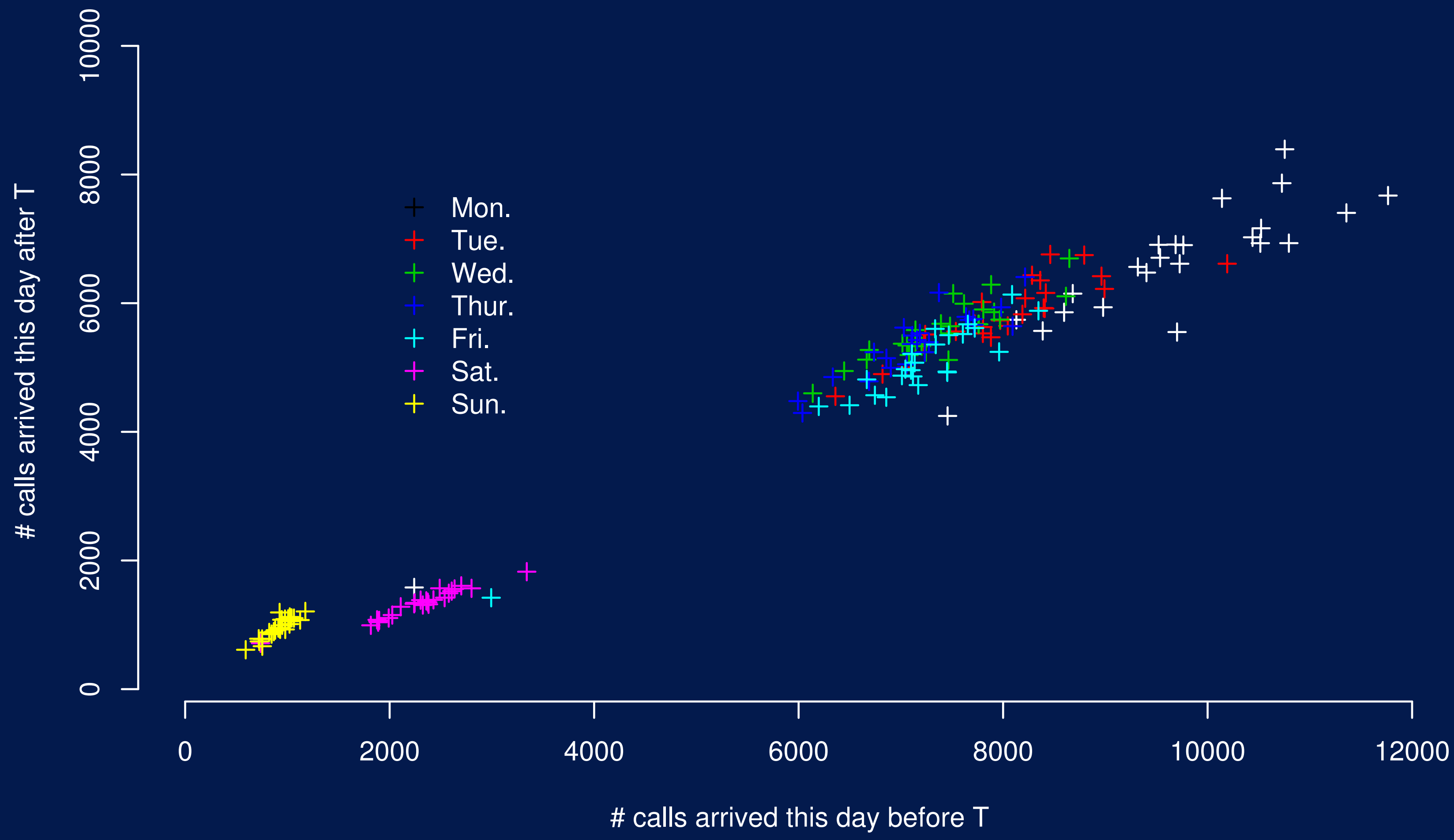
Scatter plot: (calls before T, calls after T), all days, $T = 10$ a.m.



Scatter plot, all days $T = 11$ a.m.



Scatter plot, all days $T = 12$ a.m.



Scatter plot, all days $T = 2$ p.m.

Arrival process modeling

Arrival processes can be nonstationary, doubly stochastic, dependent, etc.
Examples, for single type: nonstationary Poisson with random inflation factor each day; arrival rate $B\lambda(t)$ at time t, where $\mathbb{E}[B] = 1$.
[Whitt, Avramidis et al. 2004].

Arrival process modeling

Arrival processes can be nonstationary, doubly stochastic, dependent, etc.
Examples, for single type: nonstationary Poisson with random inflation factor each day; arrival rate $B\lambda(t)$ at time t, where $\mathbb{E}[B] = 1$.
[Whitt, Avramidis et al. 2004].

Other multivariate distributions (NORTA, etc.).

Arrival process modeling

Arrival processes can be nonstationary, doubly stochastic, dependent, etc.
Examples, for single type: nonstationary Poisson with random inflation factor each day; arrival rate $B\lambda(t)$ at time t, where $\mathbb{E}[B] = 1$.
[Whitt, Avramidis et al. 2004].

Other multivariate distributions (NORTA, etc.).

But: other relevant information can be used to forecast call volumes.
All this information should be part of our models.

Arrival process modeling

Arrival processes can be nonstationary, doubly stochastic, dependent, etc.
Examples, for single type: nonstationary Poisson with random inflation factor each day; arrival rate $B\lambda(t)$ at time t, where $\mathbb{E}[B] = 1$.
[Whitt, Avramidis et al. 2004].

Other multivariate distributions (NORTA, etc.).

But: other relevant information can be used to forecast call volumes.
All this information should be part of our models.

Service time distributions: lognormal often fits well. Sometimes loglogistic.
Patience time distributions;
Disconnections;
Etc.

Steady-state approximations of SL

Single skill: M/M/s (Erlang-C Analysis).

μ = service rate; λ = arrival rate; $r = \lambda/\mu$ = load
s = staffing (number of servers)
$\rho = \lambda/s\mu$ = utilization factor
W = steady-state delay in queue

Delay probability:

$$\mathbb{P}[W > 0] \; = \; \frac{\dfrac{r^s}{s!}\dfrac{s}{s-r}}{\dfrac{r^s}{s!}\dfrac{s}{s-r} + \displaystyle\sum_{i=0}^{s-1} \frac{r^i}{i!}}$$

Steady-state approximations of SL

Single skill: M/M/s (Erlang-C Analysis).

μ = service rate; λ = arrival rate; $r = \lambda/\mu$ = load
s = staffing (number of servers)
$\rho = \lambda/s\mu$ = utilization factor
W = steady-state delay in queue

Delay probability:

$$\mathbb{P}[W > 0] \;=\; \frac{\dfrac{r^s}{s!}\dfrac{s}{s-r}}{\dfrac{r^s}{s!}\dfrac{s}{s-r} + \displaystyle\sum_{i=0}^{s-1}\frac{r^i}{i!}}$$

Service level:

$$\mathbb{P}[W > \tau] = \mathbb{P}[W > 0]\exp[(s\mu - \lambda)\tau]$$

(conditional waiting time is exponential with mean $1/(s\mu - \lambda)$.

Large system: square-root safety staffing.

Halfin-Whitt (1981) limit: sequence of M/M/s queues where μ is fixed, $s \to \infty$, and $(1-\rho)\sqrt{s} \to \beta$ for $0 < \beta < 1$.

Theorem. Under this regime,

$$\mathbf{P}(W > 0) \;\to\; \delta = \frac{1}{1 + \beta\Phi(\beta)/\phi(\beta)}$$

where Φ and ϕ are the standard normal c.d.f. and p.d.f.

Large system: square-root safety staffing.

Halfin-Whitt (1981) limit: sequence of M/M/s queues where μ is fixed, $s \to \infty$, and $(1-\rho)\sqrt{s} \to \beta$ for $0 < \beta < 1$.

Theorem. Under this regime,

$$\mathbf{P}(W > 0) \ \to \ \delta = \frac{1}{1 + \beta\Phi(\beta)/\phi(\beta)}$$

where Φ and ϕ are the standard normal c.d.f. and p.d.f.

Square-root formula: For load r and delay probability δ, staff

$$s = r + \beta\sqrt{r}$$

agents. Simpler than inverting M/M/s formula.

The safety staffing as a fraction of the load is $\beta/\sqrt{r} \to 0$.

Other asymptotic regimes.

Conventional regime:

s is fixed, λ and μ increase linearly, $\rho \longrightarrow 1$ and $\mathbb{P}[W > 0] \longrightarrow 1$.

Other asymptotic regimes.

Conventional regime:
s is fixed, λ and μ increase linearly, $\rho \to 1$ and $\mathbb{P}[W > 0] \to 1$.

Quality-driven regime:
ρ is fixed, $\lambda \to \infty$, $s \to \infty$, then $\mathbb{P}[W > 0] \to 0$.

Other asymptotic regimes.

Conventional regime:
s is fixed, λ and μ increase linearly, $\rho \to 1$ and $\mathbb{P}[W > 0] \to 1$.

Quality-driven regime:
ρ is fixed, $\lambda \to \infty$, $s \to \infty$, then $\mathbb{P}[W > 0] \to 0$.

Efficiency-driven regime:
$s - r$ is fixed, $\lambda \to \infty$, $s \to \infty$, then $\mathbb{P}[W > 0] \to 1$.

Other asymptotic regimes.

Conventional regime:
s is fixed, λ and μ increase linearly, $\rho \to 1$ and $\mathbb{P}[W > 0] \to 1$.

Quality-driven regime:
ρ is fixed, $\lambda \to \infty$, $s \to \infty$, then $\mathbb{P}[W > 0] \to 0$.

Efficiency-driven regime:
$s - r$ is fixed, $\lambda \to \infty$, $s \to \infty$, then $\mathbb{P}[W > 0] \to 1$.

Quality and efficiency driven (QED) regime:
$\mathbb{P}[W > 0]$ is fixed in $(0, 1)$ (e.g., Halfin-Whitt).
This regime is appropriate for typical large call centers.

Models with abandonments.

These approximations have counterparts in models with Markovian abandonments.

Then the Erlang-C becomes Erlang-A.

Models with abandonments.

These approximations have counterparts in models with Markovian abandonments. Then the Erlang-C becomes Erlang-A.

Square-root formula still applies for QED regime, but with a smaller β.

Models with abandonments.

These approximations have counterparts in models with Markovian abandonments.

Then the Erlang-C becomes Erlang-A.

Square-root formula still applies for QED regime, but with a smaller β.

In the limit when mean patience times $\to 0$, we have a loss system: Erlang-B).

Time-dependent arrival rate.

Pointwise Stationary (PS) approximation [Green and Kolesar 1991, Whitt 1991]:

1. Use steady-state approximation at each time point t;
2. integrate result (performance) with respect to t.

Time-dependent arrival rate.

Pointwise Stationary (PS) approximation [Green and Kolesar 1991, Whitt 1991]:

1. Use steady-state approximation at each time point t;
2. integrate result (performance) with respect to t.

Improvement: use arrival rate at t for performance at $t + \delta$, for some delay δ [e.g., Ingolfsson et al. 2000, Green and Kolesar 1991].

Multiskill case: loss system, approximation of loss probability.

Equivalent Random Method: two-moment approximation of overflow process [Hayward 1981, Wolff 1989, Chevalier et. al. 2004].

Multiskill case: loss system, approximation of loss probability.

Equivalent Random Method: two-moment approximation of overflow process [Hayward 1981, Wolff 1989, Chevalier et. al. 2004].

Hyper-exponential approximation [Franx et. al. 2005]: Reportedly very successul at approximating by-type loss rates. Restrictions on routing.

Multiskill case: loss system, approximation of loss probability.

Equivalent Random Method: two-moment approximation of overflow process [Hayward 1981, Wolff 1989, Chevalier et. al. 2004].

Hyper-exponential approximation [Franx et. al. 2005]: Reportedly very successul at approximating by-type loss rates. Restrictions on routing.

Static overflow routing for loss system [Koole and Talim 2000]:

Call type k has a Poisson arrival process, with rate λ_k, and ordered list of agent types to look for.

Multiskill case: loss system, approximation of loss probability.

Equivalent Random Method: two-moment approximation of overflow process [Hayward 1981, Wolff 1989, Chevalier et. al. 2004].

Hyper-exponential approximation [Franx et. al. 2005]: Reportedly very successul at approximating by-type loss rates. Restrictions on routing.

Static overflow routing for loss system [Koole and Talim 2000]:

Call type k has a Poisson arrival process, with rate λ_k, and ordered list of agent types to look for.

If all these agent types are busy, the call is lost.

Multiskill case: loss system, approximation of loss probability.

Equivalent Random Method: two-moment approximation of overflow process [Hayward 1981, Wolff 1989, Chevalier et. al. 2004].

Hyper-exponential approximation [Franx et. al. 2005]: Reportedly very successul at approximating by-type loss rates. Restrictions on routing.

Static overflow routing for loss system [Koole and Talim 2000]:

Call type k has a Poisson arrival process, with rate λ_k, and ordered list of agent types to look for.

If all these agent types are busy, the call is lost.
The net arrival process at each skill group is assumed Poisson.

Multiskill case: loss system, approximation of loss probability.

Equivalent Random Method: two-moment approximation of overflow process [Hayward 1981, Wolff 1989, Chevalier et. al. 2004].

Hyper-exponential approximation [Franx et. al. 2005]: Reportedly very successul at approximating by-type loss rates. Restrictions on routing.

Static overflow routing for loss system [Koole and Talim 2000]:

Call type k has a Poisson arrival process, with rate λ_k, and ordered list of agent types to look for.

If all these agent types are busy, the call is lost.
The net arrival process at each skill group is assumed Poisson.

Blocking probabilities per skill group and loss rates per call type are computed by solving a system of nonlinear equations.

Loss-delay approximation of SL [Avramidis et al. 2006].

Loss-delay approximation of SL [Avramidis et al. 2006].

Assume customer type k joins a waiting queue at the last skill group in its list, if all are busy.

Loss-delay approximation of SL [Avramidis et al. 2006].

Assume customer type k joins a waiting queue at the last skill group in its list, if all are busy.

The SL for type k is approximated using a birth-and-death CTMC model at that queue, for each k.

Loss-delay approximation of SL [Avramidis et al. 2006].

Assume customer type k joins a waiting queue at the last skill group in its list, if all are busy.

The SL for type k is approximated using a birth-and-death CTMC model at that queue, for each k.

Allows abandonments and general routing.

Loss-delay approximation of SL [Avramidis et al. 2006].

Assume customer type k joins a waiting queue at the last skill group in its list, if all are busy.

The SL for type k is approximated using a birth-and-death CTMC model at that queue, for each k.

Allows abandonments and general routing.

Not very realistic, but useful for rough-cut first-step in staffing optimization.

Loss-delay approximation of SL [Avramidis et al. 2006].

Assume customer type k joins a waiting queue at the last skill group in its list, if all are busy.

The SL for type k is approximated using a birth-and-death CTMC model at that queue, for each k.

Allows abandonments and general routing.

Not very realistic, but useful for rough-cut first-step in staffing optimization.

For realistic models: must use simulation.

Simulation tools

Paradigm: a computer game that reproduces exactly the behavior of the call center relevant to the decisions we want to make.

Simulation tools

Paradigm: a computer game that reproduces exactly the behavior of the call center relevant to the decisions we want to make.

Then we can try everything we want without paying the price.

Simulation tools

Paradigm: a computer game that reproduces exactly the behavior of the call center relevant to the decisions we want to make.

Then we can try everything we want without paying the price.

Available software to do this?
Arena Contact Center Edition, ccProphet, ...

Simulation tools

Paradigm: a computer game that reproduces exactly the behavior of the call center relevant to the decisions we want to make.

Then we can try everything we want without paying the price.

Available software to do this?
Arena Contact Center Edition, ccProphet, ...

Plus: Graphical user interface and animation.

Simulation tools

Paradigm: a computer game that reproduces exactly the behavior of the call center relevant to the decisions we want to make.

Then we can try everything we want without paying the price.

Available software to do this?
Arena Contact Center Edition, ccProphet, ...

Plus: Graphical user interface and animation.

Minus: Expensive, slow, not so flexible.

Simulation tools

Paradigm: a computer game that reproduces exactly the behavior of the call center relevant to the decisions we want to make.

Then we can try everything we want without paying the price.

Available software to do this?
Arena Contact Center Edition, ccProphet, ...

Plus: Graphical user interface and animation.

Minus: Expensive, slow, not so flexible.

So we developed our own tool: ContactCenters, a Java library built over SSJ.

ContactCenters

Library based on well-supported modern programming language Java.
Strong expressive power. GUI-independent.
Interoperability with other software (statistics, optimization, databases, etc).

ContactCenters

Library based on well-supported modern programming language Java.
Strong expressive power. GUI-independent.
Interoperability with other software (statistics, optimization, databases, etc).

Fast: about 30 times faster than Arena in our experiments.

ContactCenters

Library based on well-supported modern programming language Java.
Strong expressive power. GUI-independent.
Interoperability with other software (statistics, optimization, databases, etc).

Fast: about 30 times faster than Arena in our experiments.

Provides building blocks to simulate all types of call centers.
Supports several types of contacts, multiskill, blend, arbitrary dialing and routing policies, various types of arrival processes, etc.

ContactCenters

Library based on well-supported modern programming language Java.
Strong expressive power. GUI-independent.
Interoperability with other software (statistics, optimization, databases, etc).

Fast: about 30 times faster than Arena in our experiments.

Provides building blocks to simulate all types of call centers.
Supports several types of contacts, multiskill, blend, arbitrary dialing and routing policies, various types of arrival processes, etc.

Support for variance reduction.

ContactCenters

Library based on well-supported modern programming language Java.
Strong expressive power. GUI-independent.
Interoperability with other software (statistics, optimization, databases, etc).

Fast: about 30 times faster than Arena in our experiments.

Provides building blocks to simulate all types of call centers.
Supports several types of contacts, multiskill, blend, arbitrary dialing and routing policies, various types of arrival processes, etc.

Support for variance reduction.

Pre-compiled generic models.

Optimization: Integer Programming Formulation

Call types	$k = 1, \ldots, K$;
Agent types (or skill groups)	$i = 1, \ldots, I$;
Periods	$p = 1, \ldots, P$;
Shift types	$q = 1, \ldots, Q$.

Optimization: Integer Programming Formulation

Call types	$k = 1, \ldots, K$;
Agent types (or skill groups)	$i = 1, \ldots, I$;
Periods	$p = 1, \ldots, P$;
Shift types	$q = 1, \ldots, Q$.

The shift specifies the time when the agent starts working, when he/she finishes, and all the lunch and coffee breaks.

Optimization: Integer Programming Formulation

Call types	$k = 1, \ldots, K$;
Agent types (or skill groups)	$i = 1, \ldots, I$;
Periods	$p = 1, \ldots, P$;
Shift types	$q = 1, \ldots, Q$.

The shift specifies the time when the agent starts working, when he/she finishes, and all the lunch and coffee breaks.

Costs: $\mathbf{c} = (c_{1,1}, \ldots, c_{1,Q}, \ldots, c_{I,1}, \ldots, c_{I,Q})$ where
$c_{i,q}$ = cost of an agent of type i having shift q.

Optimization: Integer Programming Formulation

Call types	$k = 1, \ldots, K$;
Agent types (or skill groups)	$i = 1, \ldots, I$;
Periods	$p = 1, \ldots, P$;
Shift types	$q = 1, \ldots, Q$.

The shift specifies the time when the agent starts working, when he/she finishes, and all the lunch and coffee breaks.

Costs: $\mathbf{c} = (c_{1,1}, \ldots, c_{1,Q}, \ldots, c_{I,1}, \ldots, c_{I,Q})$ where
$c_{i,q}$ = cost of an agent of type i having shift q.

Decision variables: $\mathbf{x} = (x_{1,1}, \ldots, x_{1,Q}, \ldots, x_{I,1}, \ldots, x_{I,Q})$ where
$x_{i,q}$ = number of agents of type i having shift q.

Optimization: Integer Programming Formulation

Call types	$k = 1, \ldots, K$;
Agent types (or skill groups)	$i = 1, \ldots, I$;
Periods	$p = 1, \ldots, P$;
Shift types	$q = 1, \ldots, Q$.

The shift specifies the time when the agent starts working, when he/she finishes, and all the lunch and coffee breaks.

Costs: $\mathbf{c} = (c_{1,1}, \ldots, c_{1,Q}, \ldots, c_{I,1}, \ldots, c_{I,Q})$ where
$c_{i,q}$ = cost of an agent of type i having shift q.

Decision variables: $\mathbf{x} = (x_{1,1}, \ldots, x_{1,Q}, \ldots, x_{I,1}, \ldots, x_{I,Q})$ where
$x_{i,q}$ = number of agents of type i having shift q.

Auxiliary variables: $\mathbf{y} = (y_{1,1}, \ldots, y_{1,P}, \ldots, y_{I,1}, \ldots, y_{I,P})$ where
$y_{i,p}$ = number of agents of type i in period p.

Optimization: Integer Programming Formulation

Call types	$k = 1, \ldots, K$;
Agent types (or skill groups)	$i = 1, \ldots, I$;
Periods	$p = 1, \ldots, P$;
Shift types	$q = 1, \ldots, Q$.

The shift specifies the time when the agent starts working, when he/she finishes, and all the lunch and coffee breaks.

Costs: $\mathbf{c} = (c_{1,1}, \ldots, c_{1,Q}, \ldots, c_{I,1}, \ldots, c_{I,Q})$ where
$c_{i,q}$ = cost of an agent of type i having shift q.

Decision variables: $\mathbf{x} = (x_{1,1}, \ldots, x_{1,Q}, \ldots, x_{I,1}, \ldots, x_{I,Q})$ where
$x_{i,q}$ = number of agents of type i having shift q.

Auxiliary variables: $\mathbf{y} = (y_{1,1}, \ldots, y_{1,P}, \ldots, y_{I,1}, \ldots, y_{I,P})$ where
$y_{i,p}$ = number of agents of type i in period p.

We have $\mathbf{y} = \mathbf{A}\mathbf{x}$ where $\mathbf{A}$ is block diagonal with i blocks $\tilde{\mathbf{A}}$, and element (p, q) of $\tilde{\mathbf{A}}$ is 1 if shift q covers period p, 0 otherwise.

$g_{k,p}(\mathbf{y})$ = long-run SL for call type k in period p.
E.g., fraction of calls answered within $\tau_{k,p}$ seconds:

$$g_{k,p}(\mathbf{y}) = \frac{E[\text{num. of calls in period } p \text{ answered within the limit}]}{E[\text{num. of arrivals in period } p]}.$$

$g_{k,p}(\mathbf{y})$ = long-run SL for call type k in period p.
E.g., fraction of calls answered within $\tau_{k,p}$ seconds:

$$g_{k,p}(\mathbf{y}) = \frac{E[\text{num. of calls in period } p \text{ answered within the limit}]}{E[\text{num. of arrivals in period } p]}.$$

$g_p(\mathbf{y})$ = aggregated long-run SL over period p.
$g_k(\mathbf{y})$ = aggregated long-run SL for call type k.
$g(\mathbf{y})$ = aggregated long-run SL overall.

$g_{k,p}(\mathbf{y})$ = long-run SL for call type k in period p.
E.g., fraction of calls answered within $\tau_{k,p}$ seconds:

$$g_{k,p}(\mathbf{y}) = \frac{E[\text{num. of calls in period } p \text{ answered within the limit}]}{E[\text{num. of arrivals in period } p]}.$$

$g_p(\mathbf{y})$ = aggregated long-run SL over period p.
$g_k(\mathbf{y})$ = aggregated long-run SL for call type k.
$g(\mathbf{y})$ = aggregated long-run SL overall.

These functions $g_\bullet$ (ratios of expectations) are unknown and often very complicated. Each may depend on entire vector $\mathbf{y}$. They can be either

- approximated via simplified queueing models, or
- estimated by simulation.

Here, the routing rules are assumed fixed; we do not optimize them.
(But eventually we should.)

Scheduling problem

$$
\begin{aligned}
\min \quad & \mathbf{c}^{\mathrm{t}}\mathbf{x} = \sum_{i=1}^{I} \sum_{q=1}^{Q} c_{i,q} x_{i,q} \\
\text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{y}, \\
& g_{k,p}(\mathbf{y}) \geq l_{k,p} \qquad \text{for all } k, p, \\
& g_p(\mathbf{y}) \geq l_p \qquad \text{for all } p, \\
& g_k(\mathbf{y}) \geq l_k \qquad \text{for all } k, \\
& g(\mathbf{y}) \geq l, \\
& \mathbf{x} \geq 0, \ \text{and integer.}
\end{aligned}
$$

Staffing Problem

Relaxation: forget about the admissibility of schedules; just assume that any staffing is admissible.

Staffing Problem

Relaxation: forget about the admissibility of schedules; just assume that any staffing is admissible.

Costs: $\mathbf{c} = (c_{1,1}, \ldots, c_{1,P}, \ldots, c_{I,1}, \ldots, c_{I,P})$ where
$c_{i,p}$ = cost of an agent of type i in period p.

Staffing Problem

Relaxation: forget about the admissibility of schedules; just assume that any staffing is admissible.

Costs: $\mathbf{c} = (c_{1,1}, \ldots, c_{1,P}, \ldots, c_{I,1}, \ldots, c_{I,P})$ where
$c_{i,p}$ = cost of an agent of type i in period p.

$$
\begin{aligned}
\min \quad & \mathbf{c}^{\mathsf{t}}\mathbf{y} = \sum_{i=1}^{I} \sum_{p=1}^{P} c_{i,p} y_{i,p} \\
\text{subject to} \quad & g_{k,p}(\mathbf{y}) \ge l_{k,p} \qquad \text{for all } k, p, \\
& g_p(\mathbf{y}) \ge l_p \qquad \text{for all } p, \\
& g_k(\mathbf{y}) \ge l_k \qquad \text{for all } k, \\
& g(\mathbf{y}) \ge l, \\
& \mathbf{y} \ge 0, \ \text{and integer.}
\end{aligned}
$$

Single period, steady-state approximation

Take one period at a time and assume that the system is in steady-state.

Single period, steady-state approximation

Take one period at a time and assume that the system is in steady-state.

$\mathbf{c} = (c_1, \ldots, c_I)$ where c_i = cost of an agent of type i.

Single period, steady-state approximation

Take one period at a time and assume that the system is in steady-state.

$\mathbf{c} = (c_1, \ldots, c_I)$ where c_i = cost of an agent of type i.

$\mathbf{y} = (y_1, \ldots, y_I)$ where y_i = number of agents of type i.

Single period, steady-state approximation

Take one period at a time and assume that the system is in steady-state.

$\mathbf{c} = (c_1, \ldots, c_I)$ where c_i = cost of an agent of type i.

$\mathbf{y} = (y_1, \ldots, y_I)$ where y_i = number of agents of type i.

$$
\begin{array}{rll}
\min & \mathbf{c}^{\mathrm{t}}\mathbf{y} = \sum_{i=1}^{I} c_i y_i & \\
\text{subject to} & g_k(\mathbf{y}) \geq l_k & \text{for all } k, \\
& g(\mathbf{y}) \geq l, & \\
& \mathbf{y} \geq 0, \text{ and integer.} &
\end{array}
$$

Single period, steady-state approximation

Take one period at a time and assume that the system is in steady-state.

$\mathbf{c} = (c_1, \ldots, c_I)$ where c_i = cost of an agent of type i.

$\mathbf{y} = (y_1, \ldots, y_I)$ where y_i = number of agents of type i.

$$\begin{array}{rll} \min & \mathbf{c}^{\mathrm{t}}\mathbf{y} = \sum_{i=1}^{I} c_i y_i & \\ \text{subject to} & g_k(\mathbf{y}) \geq l_k & \text{for all } k, \\ & g(\mathbf{y}) \geq l, & \\ & \mathbf{y} \geq 0, \text{ and integer.} & \end{array}$$

Neglects the transient and overflow effect across periods.

Scheduling by a two-step method

First step: solve the staffing problem (easier).
Let $\mathbf{y}^*$ be an optimal solution.

Scheduling by a two-step method

First step: solve the staffing problem (easier).
Let $\mathbf{y}^*$ be an optimal solution.

Second step: find an admissible schedule that covers the staffing $\mathbf{y}^*$, by solving:

$$\begin{aligned} \min \quad & \mathbf{c}^{\mathrm{t}}\mathbf{x} = \sum_{i=1}^{I} \sum_{q=1}^{Q} c_{i,q} x_{i,q} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{y}^*, \\ & \mathbf{x} \geq 0, \text{ and integer.} \end{aligned}$$

Scheduling by a two-step method

First step: solve the staffing problem (easier).
Let $\mathbf{y}^*$ be an optimal solution.

Second step: find an admissible schedule that covers the staffing $\mathbf{y}^*$, by solving:

$$
\begin{aligned}
\min \quad & \mathbf{c}^{\mathrm{t}}\mathbf{x} = \sum_{i=1}^{I} \sum_{q=1}^{Q} c_{i,q} x_{i,q} \\
\text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{y}^*, \\
& \mathbf{x} \geq 0, \ \text{and integer.}
\end{aligned}
$$

This does not give an optimal solution to the original scheduling problem.
And the gap could be significant.

Scheduling by a two-step method

First step: solve the staffing problem (easier).
Let $\mathbf{y}^*$ be an optimal solution.

Second step: find an admissible schedule that covers the staffing $\mathbf{y}^*$, by solving:

$$
\begin{aligned}
\min \quad & \mathbf{c}^{\mathrm{t}}\mathbf{x} = \sum_{i=1}^{I} \sum_{q=1}^{Q} c_{i,q} x_{i,q} \\
\text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{y}^*, \\
& \mathbf{x} \geq 0, \ \text{and integer.}
\end{aligned}
$$

This does not give an optimal solution to the original scheduling problem.
And the gap could be significant.
Potential discrepancy in mix of agent types across successive periods.

Scheduling by a two-step method

First step: solve the staffing problem (easier).
Let $\mathbf{y}^*$ be an optimal solution.

Second step: find an admissible schedule that covers the staffing $\mathbf{y}^*$, by solving:

$$
\begin{aligned}
\min \quad & \mathbf{c}^{\mathrm{t}}\mathbf{x} = \sum_{i=1}^{I} \sum_{q=1}^{Q} c_{i,q} x_{i,q} \\
\text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{y}^*, \\
& \mathbf{x} \geq 0, \text{ and integer.}
\end{aligned}
$$

This does not give an optimal solution to the original scheduling problem.
And the gap could be significant.
Potential discrepancy in mix of agent types across successive periods.
Can be alleviated by skill-transfer decision variables.

Skill-transfer decision variables [Bhulai, Koole, and Pot 2005]:
$z_{p,j,i}$ = number of agents of type j that work as type-i agents in period p (they use only part of their skills).

$$\min \quad \mathbf{c}^{\mathrm{t}}\mathbf{x} = \sum_{i=1}^{I}\sum_{q=1}^{Q} c_{i,q}x_{i,q}$$

subject to

$$\sum_{q:\tilde{A}_{p,q}=1} x_{i,q} + \sum_{j\in\mathcal{S}_i^+} z_{p,j,i} - \sum_{j\in\mathcal{S}_i^-} z_{p,i,j} \geq y_{i,p} \quad \forall p,\ i$$

all $x_{i,q}, z_{p,j,i} \geq 0$ and integer.

Sample-path optimization via simulation

[Atlason, Epelman, and Henderson, 2004; Cezik and L'Ecuyer 2005]
We simulate n independent operating days (could also be weeks, etc.) of the center, to estimate the functions $g_\bullet$.

Sample-path optimization via simulation

[Atlason, Epelman, and Henderson, 2004; Cezik and L'Ecuyer 2005]
We simulate n independent operating days (could also be weeks, etc.) of the center, to estimate the functions $g_\bullet$.

Let ω represent the source of randomness, i.e., the sequence of independent uniform r.v.'s underlying the entire simulation (n runs).

Sample-path optimization via simulation

[Atlason, Epelman, and Henderson, 2004; Cezik and L'Ecuyer 2005]
We simulate n independent operating days (could also be weeks, etc.) of the center, to estimate the functions $g_\bullet$.

Let ω represent the source of randomness, i.e., the sequence of independent uniform r.v.'s underlying the entire simulation (n runs).

For a ω, the empirical SL's over the n simulation runs are:

$G_{n,k,p}(\mathbf{y}, \omega)$ for call type k in period p;
$G_{n,p}(\mathbf{y}, \omega)$ aggregated over period p;
$G_{n,k}(\mathbf{y}, \omega)$ aggregated for call type k;
$G_n(\mathbf{y}, \omega)$ aggregated overall.

Sample-path optimization via simulation

[Atlason, Epelman, and Henderson, 2004; Cezik and L'Ecuyer 2005]
We simulate n independent operating days (could also be weeks, etc.) of the center, to estimate the functions $g_\bullet$.

Let ω represent the source of randomness, i.e., the sequence of independent uniform r.v.'s underlying the entire simulation (n runs).

For a ω, the empirical SL's over the n simulation runs are:
$G_{n,k,p}(\mathbf{y}, \omega)$ for call type k in period p;
$G_{n,p}(\mathbf{y}, \omega)$ aggregated over period p;
$G_{n,k}(\mathbf{y}, \omega)$ aggregated for call type k;
$G_n(\mathbf{y}, \omega)$ aggregated overall.

For a fixed ω, these are deterministic functions of $\mathbf{y}$.

Sample-path optimization via simulation

[Atlason, Epelman, and Henderson, 2004; Cezik and L'Ecuyer 2005]
We simulate n independent operating days (could also be weeks, etc.) of the center, to estimate the functions $g_\bullet$.

Let ω represent the source of randomness, i.e., the sequence of independent uniform r.v.'s underlying the entire simulation (n runs).

For a ω, the empirical SL's over the n simulation runs are:
$G_{n,k,p}(\mathbf{y}, \omega)$ for call type k in period p;
$G_{n,p}(\mathbf{y}, \omega)$ aggregated over period p;
$G_{n,k}(\mathbf{y}, \omega)$ aggregated for call type k;
$G_n(\mathbf{y}, \omega)$ aggregated overall.

For a fixed ω, these are deterministic functions of $\mathbf{y}$.

We replace the functions g by the estimators G and optimize.

Sample-path optimization via simulation

[Atlason, Epelman, and Henderson, 2004; Cezik and L'Ecuyer 2005]
We simulate n independent operating days (could also be weeks, etc.) of the center, to estimate the functions $g_\bullet$.

Let ω represent the source of randomness, i.e., the sequence of independent uniform r.v.'s underlying the entire simulation (n runs).

For a ω, the empirical SL's over the n simulation runs are:

$G_{n,k,p}(\mathbf{y},\omega)$ for call type k in period p;
$G_{n,p}(\mathbf{y},\omega)$ aggregated over period p;
$G_{n,k}(\mathbf{y},\omega)$ aggregated for call type k;
$G_n(\mathbf{y},\omega)$ aggregated overall.

For a fixed ω, these are deterministic functions of $\mathbf{y}$.

We replace the functions g by the estimators G and optimize.

To compute them at different values of $\mathbf{y}$, we simply use simulation with well-synchronized common random numbers.

Empirical scheduling optimization problem (sample version of the problem):

$$\begin{array}{rll} \min & \mathbf{c}^{\mathrm{t}}\mathbf{x} = \sum_{i=1}^{I}\sum_{q=1}^{Q} c_{i,q}x_{i,q} & \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{y}, & \\ & G_{n,k,p}(\mathbf{y}) \geq l_{k,p} & \text{for all } k, p, \\ & G_{n,p}(\mathbf{y}) \geq l_{p} & \text{for all } p, \\ & G_{n,k}(\mathbf{y}) \geq l_{k} & \text{for all } k, \\ & G_{n}(\mathbf{y}) \geq l, & \\ & \mathbf{x} \geq 0, \text{ and integer.} & \end{array}$$

Empirical scheduling optimization problem (sample version of the problem):

$$
\begin{aligned}
\min \quad & \mathbf{c}^{\mathrm{t}}\mathbf{x} = \textstyle\sum_{i=1}^{I} \sum_{q=1}^{Q} c_{i,q} x_{i,q} \\
\text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{y}, \\
& G_{n,k,p}(\mathbf{y}) \geq l_{k,p} \qquad \text{for all } k, p, \\
& G_{n,p}(\mathbf{y}) \geq l_{p} \qquad \text{for all } p, \\
& G_{n,k}(\mathbf{y}) \geq l_{k} \qquad \text{for all } k, \\
& G_{n}(\mathbf{y}) \geq l, \\
& \mathbf{x} \geq 0, \text{ and integer.}
\end{aligned}
$$

Under mild conditions, when $n \to \infty$, the optimal solution of the sample problem converges w.p.1 to that of the original problem.

Empirical scheduling optimization problem (sample version of the problem):

$$
\begin{array}{ll}
\min & \mathbf{c}^{\mathsf{t}}\mathbf{x} = \sum_{i=1}^{I}\sum_{q=1}^{Q} c_{i,q}x_{i,q} \\
\text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{y}, \\
& G_{n,k,p}(\mathbf{y}) \geq l_{k,p} \quad \text{for all } k, p, \\
& G_{n,p}(\mathbf{y}) \geq l_p \quad \text{for all } p, \\
& G_{n,k}(\mathbf{y}) \geq l_k \quad \text{for all } k, \\
& G_n(\mathbf{y}) \geq l, \\
& \mathbf{x} \geq 0, \text{ and integer.}
\end{array}
$$

Under mild conditions, when $n \to \infty$, the optimal solution of the sample problem converges w.p.1 to that of the original problem.

Similar formulation for the staffing problem.

Linear / integer programming with cutting planes

[Atlason, Epelman, and Henderson, 2004; Cezik and L'Ecuyer 2005]

Start with a relaxation of the IP and add cuts (linear constraints) derived from the SL constraints that are not satisfied in the sample problem.
Stop when all SL constraints are satisfied.

Linear / integer programming with cutting planes

[Atlason, Epelman, and Henderson, 2004; Cezik and L'Ecuyer 2005]

Start with a relaxation of the IP and add cuts (linear constraints) derived from the SL constraints that are not satisfied in the sample problem.
Stop when all SL constraints are satisfied.

Then run longer simulation to perform a local adjustment.

Linear / integer programming with cutting planes

[Atlason, Epelman, and Henderson, 2004; Cezik and L'Ecuyer 2005]

Start with a relaxation of the IP and add cuts (linear constraints) derived from the SL constraints that are not satisfied in the sample problem.
Stop when all SL constraints are satisfied.

Then run longer simulation to perform a local adjustment.

Initial constraints are obtained by imposing that skill supply covers the load.

Linear / integer programming with cutting planes

[Atlason, Epelman, and Henderson, 2004; Cezik and L'Ecuyer 2005]

Start with a relaxation of the IP and add cuts (linear constraints) derived from the SL constraints that are not satisfied in the sample problem.
Stop when all SL constraints are satisfied.

Then run longer simulation to perform a local adjustment.

Initial constraints are obtained by imposing that skill supply covers the load.

A realistic staffing problem with 65 call types and 89 agent types was solved (approx.) by this approach.

Heuristic neighborhood search for staffing problem

[Avramidis et al. 2006]

Stage 1: Start with feasible solution and decrease cost using neighborhood search with “loss-delay” approximation to evaluate SL and select next solution.

Heuristic neighborhood search for staffing problem

[Avramidis et al. 2006]
Stage 1: Start with feasible solution and decrease cost using neighborhood search with "loss-delay" approximation to evaluate SL and select next solution.

Outputs a locally optimal solution (w.r.t. loss-delay approx.).

Heuristic neighborhood search for staffing problem

[Avramidis et al. 2006]

Stage 1: Start with feasible solution and decrease cost using neighborhood search with "loss-delay" approximation to evaluate SL and select next solution.

Outputs a locally optimal solution (w.r.t. loss-delay approx.).

Stage 2: simulation-based local adjustment to stage-1 solution to account for two mutually exclusive cases:

- the incumbent is infeasible and we seek a nearby feasible solution;
- the incumbent is feasible and we seek further cost reduction.

Heuristic neighborhood search for staffing problem

[Avramidis et al. 2006]

Stage 1: Start with feasible solution and decrease cost using neighborhood search with "loss-delay" approximation to evaluate SL and select next solution.

Outputs a locally optimal solution (w.r.t. loss-delay approx.).

Stage 2: simulation-based local adjustment to stage-1 solution to account for two mutually exclusive cases:

- the incumbent is infeasible and we seek a nearby feasible solution;
- the incumbent is feasible and we seek further cost reduction.

This approach is competitive with the previous one, especially when the computing budget is small (according to our experiments).

Still more to do....

Scheduling problem:
We are getting promising results with LP and cutting planes.
Better than two-stage method and much better than neighborhood search.

Still more to do....

Scheduling problem:
We are getting promising results with LP and cutting planes.
Better than two-stage method and much better than neighborhood search.

Other constraints and objectives:
abandonment ratios, occupancy of agents, fairness across call types, etc.

Still more to do....

Scheduling problem:
We are getting promising results with LP and cutting planes.
Better than two-stage method and much better than neighborhood search.

Other constraints and objectives:
abandonment ratios, occupancy of agents, fairness across call types, etc.

Gap with real-life problem:
Call center managers use various recourses to deal with unanticipated events.
So far, we do not take that into account, but we should.

Still more to do....

Scheduling problem:
We are getting promising results with LP and cutting planes.
Better than two-stage method and much better than neighborhood search.

Other constraints and objectives:
abandonment ratios, occupancy of agents, fairness across call types, etc.

Gap with real-life problem:
Call center managers use various recourses to deal with unanticipated events.
So far, we do not take that into account, but we should.

We have also assumed fixed routing rules, no retrials, etc.

Simulation efficiency improvement

When simulating the system with several very similar values of $\mathbf{y}$ (e.g., to estimate subgradients), try to recycle the similar pieces of sample paths.

Simulation efficiency improvement

When simulating the system with several very similar values of $\mathbf{y}$ (e.g., to estimate subgradients), try to recycle the similar pieces of sample paths.

Use splitting for the more critical time periods.

Simulation efficiency improvement

When simulating the system with several very similar values of $\mathbf{y}$ (e.g., to estimate subgradients), try to recycle the similar pieces of sample paths.

Use splitting for the more critical time periods.

Stratify on most important variables (e.g., B).

Simulation efficiency improvement

When simulating the system with several very similar values of $\mathbf{y}$ (e.g., to estimate subgradients), try to recycle the similar pieces of sample paths.

Use splitting for the more critical time periods.

Stratify on most important variables (e.g., B).

Use control variates.

Variance reduction: examples

Suppose the arrival process is Poisson with rate $B\lambda(t)$ at time t, where $\mathbb{E}[B] = 1$ and $\lambda(\cdot)$ is deterministic.

Variance reduction: examples

Suppose the arrival process is Poisson with rate $B\lambda(t)$ at time t, where $\mathbb{E}[B] = 1$ and $\lambda(\cdot)$ is deterministic.

$X_i = G_i(s_0) =$ number of calls who waited less than s_0 seconds on day i. Crude estimator of $\mu = \mathbb{E}[X_i]$:

$$\bar{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i.$$

Variance reduction: examples

Suppose the arrival process is Poisson with rate $B\lambda(t)$ at time t, where $\mathbb{E}[B] = 1$ and $\lambda(\cdot)$ is deterministic.

$X_i = G_i(s_0) =$ number of calls who waited less than s_0 seconds on day i. Crude estimator of $\mu = \mathbb{E}[X_i]$:

$$\bar{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i.$$

The realization B_i of B has a large impact on $G(s_0)$, so we may want to use quasi-Monte Carlo or stratification w.r.t. B.

Variance reduction: examples

Suppose the arrival process is Poisson with rate $B\lambda(t)$ at time t, where $\mathbb{E}[B] = 1$ and $\lambda(\cdot)$ is deterministic.

$X_i = G_i(s_0) =$ number of calls who waited less than s_0 seconds on day i. Crude estimator of $\mu = \mathbb{E}[X_i]$:

$$\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i.$$

The realization B_i of B has a large impact on $G(s_0)$, so we may want to use quasi-Monte Carlo or stratification w.r.t. B.

Suppose $B = F_{\mathrm{B}}^{-1}(U)$ where $U \sim$ Unif$(0, 1)$.

Variance reduction: examples

Suppose the arrival process is Poisson with rate $B\lambda(t)$ at time t, where $\mathbb{E}[B] = 1$ and $\lambda(\cdot)$ is deterministic.

$X_i = G_i(s_0) =$ number of calls who waited less than s_0 seconds on day i. Crude estimator of $\mu = \mathbb{E}[X_i]$:

$$\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i.$$

The realization B_i of B has a large impact on $G(s_0)$, so we may want to use quasi-Monte Carlo or stratification w.r.t. B.

Suppose $B = F_{\mathrm{B}}^{-1}(U)$ where $U \sim \mathrm{Unif}(0, 1)$.

Stratification on U in k strata: for $s = 1, \ldots, k$, generate n_s indep. r.v.'s $U^{(s)} \sim \mathrm{Unif}((s-1)/k,\ s/k)$ and denote $X_{s,1}, \ldots, X_{s,n_s}$ the corresponding observations of $G(s_0)$.

Stratified estimator:

$$\bar{X}_{\mathrm{s},n} = \frac{1}{k}\sum_{s=1}^{k}\frac{1}{n_s}\sum_{i=1}^{n_s} X_{s,i}.$$

Stratified estimator:

$$\bar{X}_{\mathrm{s},n} = \frac{1}{k}\sum_{s=1}^{k}\frac{1}{n_s}\sum_{i=1}^{n_s} X_{s,i}.$$

The variance

$$\mathrm{Var}[\bar{X}_{\mathrm{s},n}] = \frac{1}{k^2}\sum_{s=1}^{k}\sigma_s^2/n_s$$

where $\sigma_s^2 = \mathrm{Var}[X \mid S = s]$ (must be estimated), is minimized by taking (if we neglect rounding)

$$n_s = n_s^* = n\frac{\sigma_s}{\sum_{l=1}^{k}\sigma_l/k}.$$

Stratified estimator:

$$\bar{X}_{\mathrm{s},n} = \frac{1}{k}\sum_{s=1}^{k}\frac{1}{n_s}\sum_{i=1}^{n_s} X_{s,i}.$$

The variance

$$\mathrm{Var}[\bar{X}_{\mathrm{s},n}] = \frac{1}{k^2}\sum_{s=1}^{k}\sigma_s^2/n_s$$

where $\sigma_s^2 = \mathrm{Var}[X \mid S = s]$ (must be estimated), is minimized by taking (if we neglect rounding)

$$n_s = n_s^* = n\frac{\sigma_s}{\sum_{l=1}^{k}\sigma_l/k}.$$

We have

$$\mathrm{Var}[\bar{X}_n] \quad = \quad \mathrm{Var}[\bar{X}_{\mathrm{so},n}] + \frac{1}{nk}\sum_{s=1}^{k}(\sigma_s - \bar{\sigma})^2 + \frac{1}{nk}\sum_{s=1}^{k}(\mu_s - \mu)^2.$$

Combining with a control variate.

Let A = number of arrivals during the day. CV estimator conditional on $U = u$:

$$X_{\rm c}(u) = X - \beta(u)[A - \mathbb{E}[A \mid U = u]] = X - \beta(u)C(u).$$

Combining with a control variate.

Let A = number of arrivals during the day. CV estimator conditional on $U = u$:

$$X_c(u) = X - \beta(u)[A - \mathbb{E}[A \mid U = u]] = X - \beta(u)C(u).$$

Optimal CV coefficient depend on u:

$$\beta^*(u) = \frac{\mathrm{Cov}[A, X|U = u]}{\mathrm{Var}[A|U = u]} = \frac{E[C \cdot X \mid U = u]}{E[C^2 \mid U = u]}.$$

Combining with a control variate.

Let A = number of arrivals during the day. CV estimator conditional on $U = u$:

$$X_{\mathrm{c}}(u) = X - \beta(u)[A - \mathbb{E}[A \mid U = u]] = X - \beta(u)C(u).$$

Optimal CV coefficient depend on u:

$$\beta^*(u) = \frac{\mathrm{Cov}[A, X|U = u]}{\mathrm{Var}[A|U = u]} = \frac{E[C \cdot X \mid U = u]}{E[C^2 \mid U = u]}.$$

Can approximate β^* by approximating the two functions $q_1(u) = E[C \cdot X \mid U = u]$ and $q_2(u) = E[C^2 \mid U = u]$.

Combining with a control variate.

Let $A =$ number of arrivals during the day. CV estimator conditional on $U = u$:

$$X_c(u) = X - \beta(u)[A - \mathbb{E}[A \mid U = u]] = X - \beta(u)C(u).$$

Optimal CV coefficient depend on u:

$$\beta^*(u) = \frac{\mathrm{Cov}[A, X|U = u]}{\mathrm{Var}[A|U = u]} = \frac{E[C \cdot X \mid U = u]}{E[C^2 \mid U = u]}.$$

Can approximate β^* by approximating the two functions $q_1(u) = E[C \cdot X \mid U = u]$ and $q_2(u) = E[C^2 \mid U = u]$.

Use pilot runs and least squares polynomial or spline approximation.

Combining with a control variate.

Let A = number of arrivals during the day. CV estimator conditional on $U = u$:

$$X_{\rm c}(u) = X - \beta(u)[A - \mathbb{E}[A \mid U = u]] = X - \beta(u)C(u).$$

Optimal CV coefficient depend on u:

$$\beta^*(u) = \frac{\mathrm{Cov}[A, X|U = u]}{\mathrm{Var}[A|U = u]} = \frac{E[C \cdot X \mid U = u]}{E[C^2 \mid U = u]}.$$

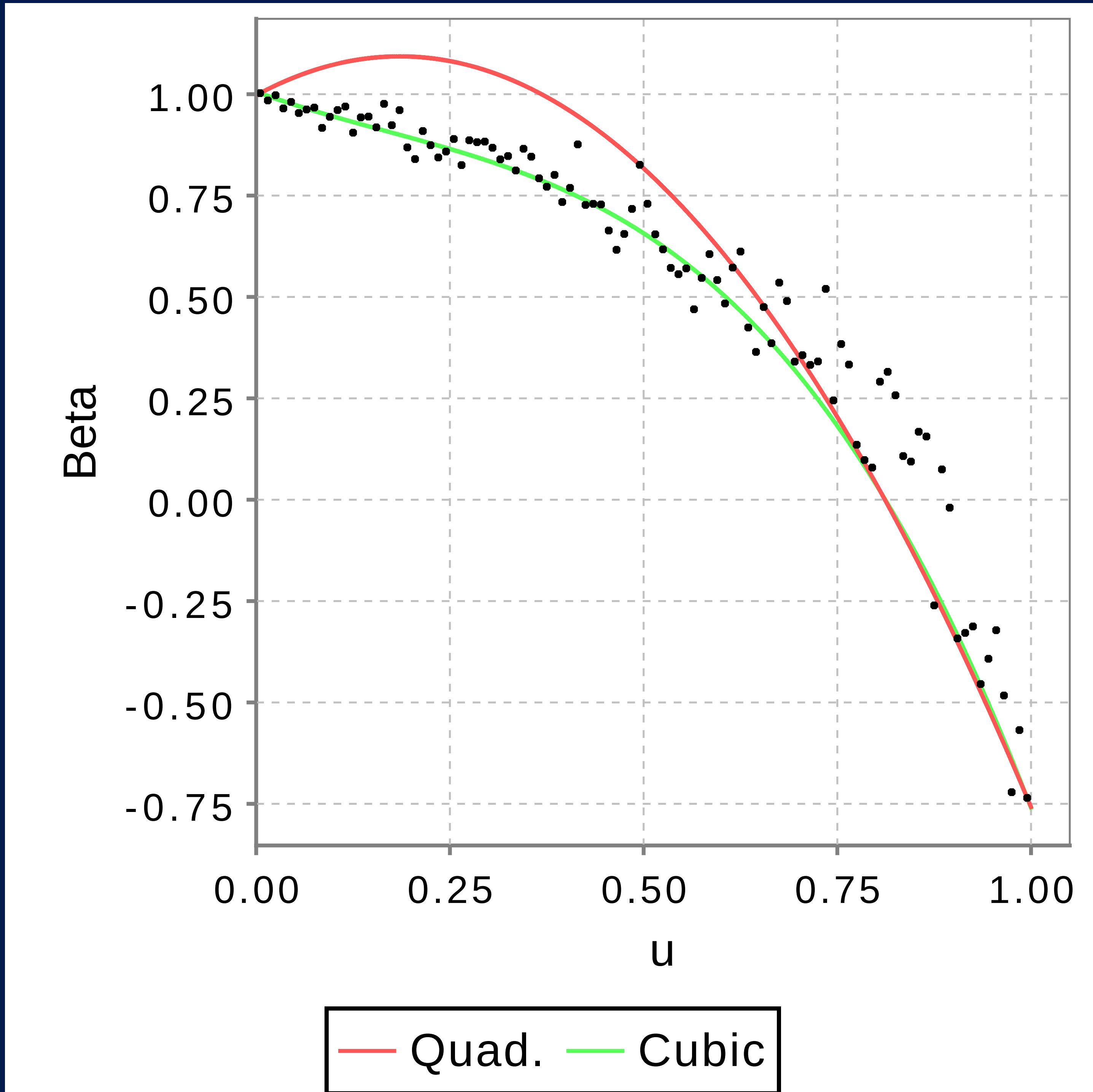
Can approximate β^* by approximating the two functions $q_1(u) = E[C \cdot X \mid U = u]$ and $q_2(u) = E[C^2 \mid U = u]$.

Use pilot runs and least squares polynomial or spline approximation.

Variance in stratum s is now

$$\sigma_s^2 = \mathrm{Var}[X_{\rm c}(U^{(s)})] = \frac{1}{k}\int_{(s-1)/k}^{s/k} \mathrm{Var}[X - \beta^*(u)C \mid U = u]du$$

where $U^{(s)} \sim$ Unif($(s-1)/k,\ s/k$). So the optimal allocation changes!



The function $\beta^*(u)$ approximated by polynomial interpolation

Another example: Sensitivity w.r.t. mean service time parameter.

Another example: Sensitivity w.r.t. mean service time parameter.

Let $X_{1,i}$ and $X_{2,i}$ be the realizations of $G(s_0)$ with mean service time θ and $\theta + \delta$.

Another example: Sensitivity w.r.t. mean service time parameter.

Let $X_{1,i}$ and $X_{2,i}$ be the realizations of $G(s_0)$ with mean service time θ and $\theta + \delta$.

Finite difference estimator:

$$\Delta_i = [X_{2,i} - X_{1,i}]/\delta.$$

Another example: Sensitivity w.r.t. mean service time parameter.

Let $X_{1,i}$ and $X_{2,i}$ be the realizations of $G(s_0)$ with mean service time θ and $\theta + \delta$.

Finite difference estimator:

$$\Delta_i = [X_{2,i} - X_{1,i}]/\delta.$$

With IRN: $\mathrm{Var}[\Delta_i] = O(\delta^{-2})$.

Another example: Sensitivity w.r.t. mean service time parameter.

Let $X_{1,i}$ and $X_{2,i}$ be the realizations of $G(s_0)$ with mean service time θ and $\theta + \delta$.

Finite difference estimator:

$$\Delta_i = [X_{2,i} - X_{1,i}]/\delta.$$

With IRN: $\mathrm{Var}[\Delta_i] = O(\delta^{-2})$.

With CRN, if $G(s_0)$ was continuous, we would have $\mathrm{Var}[\Delta_i] = O(1)$.

Another example: Sensitivity w.r.t. mean service time parameter.

Let $X_{1,i}$ and $X_{2,i}$ be the realizations of $G(s_0)$ with mean service time θ and $\theta + \delta$.

Finite difference estimator:

$$\Delta_i = [X_{2,i} - X_{1,i}]/\delta.$$

With IRN: $\mathrm{Var}[\Delta_i] = O(\delta^{-2})$.

With CRN, if $G(s_0)$ was continuous, we would have $\mathrm{Var}[\Delta_i] = O(1)$.

With well-synchronized CRN, we can prove that $\mathrm{Var}[\Delta_i] = O(\delta^{-1-\epsilon})$.

Ongoing and Future Work in Our Group

- Develop a Java library for simulation of call centers.
- Variance reduction methods for simulation.
- Integrate with optimization algorithms for staffing, scheduling, rostering, routing.
- Integrate with approximation formulas and CTMC models.
- Better models for inbound traffic and other aspects.
- Currently, our work is driven by the interests of Bell Canada.