# Simulación

Dpto. Ingeniería Industrial, Universidad de Chile

IN47B, Ingeniería de Operaciones

30 de septiembre de 2013

# Contenidos

# ¿Qué es Simulación

- Representación de un *sistema* en un computador.
- Intenta emular el funcionamiento de un *sistema*.
- Usado para evaluar *numéricamente* el comportamiento de sistemas bajo ciertas condiciones.

# ¿Qué es un Sistema?

## Definición

Un *Sistema* es un conjunto de entidades (por ejemplo máquinas o personas), que interactúan para lograr algún fin lógico.

## Estado

Es un conjunto de variables que describen un sistema en un tiempo particular.

# Ejemplo:

## Atención de clientes en un Banco

- El *sistema* puede ser el conjunto de clientes, cajeros, colas, y procedimientos predefinidos (FIFO, etc.) que describen la operación.
- Las *variables de estado* en este caso son el número de clientes en cada cola, el número de cajeros, y el estado de cada cajero (ocupado/desocupado).,

# Sistemas Discretos y Continuos

## Sistema Discreto

Un Sistema se dice *Discreto* si sus variables de estado cambian sus valores en un conjunto numerable de instantes de tiempo (Ej.: sistemas de colas).

## Sistema Continuo

Un Sistema se dice *continuo* si sus variables de estado cambian continuamente en el tiempo (Ej.: El sistema solar).

# Sistemas Dinámicos y Estáticos

## Sistema Estático

Un Sistema se dice *estático* si el tiempo en el no juega ningún rol (Ej.: Simulaciones tipo montecarlo, estimar $\pi$).

## Sistemas Dinámicos

Un Sistema se dice *dinámico* si este evoluciona a medida que el tiempo pasa.

# Sistemas Determinísticos y Estocásticos

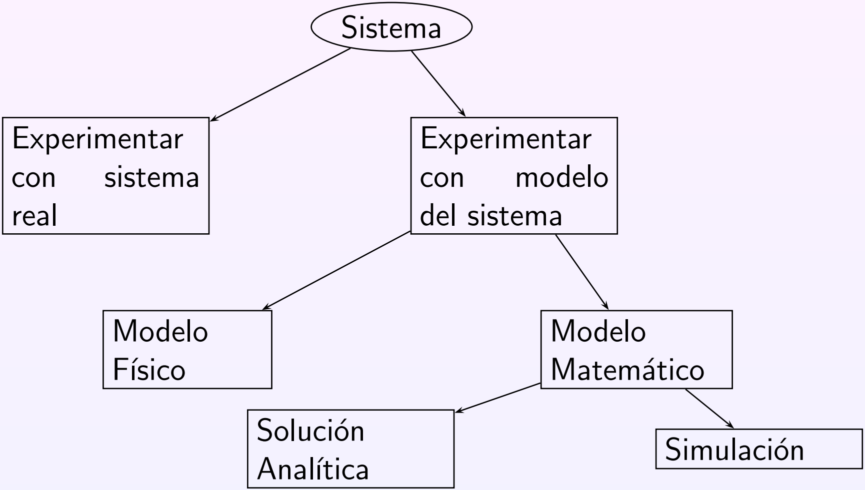
## Sistema Determinístico

Un Sistema se dice *determinista* si su evolución temporal esta definida completamente por las condiciones iniciales del mismo (Ej.: Sistema de ecuaciones diferenciales).

## Sistema Estocástico

Un Sistema se dice *estocástico* si tiene componentes descritos en términos probabilísticos, o donde existe incertidumbre en la entrada o en el proceso mismo del sistema (Ej.: Sistema de colas en un banco).

# Formas de estudiar un Sistema



# Simulación de eventos discretos

- Se asume modelo dinámico que cambia variables de estado una cantidad numerable de veces.
- Un evento es un acontecimiento instantáneo que puede cambiar el estado del sistema.
- Ejemplo M/M/1:
    - Variables de estado son largo de cola (tiempo de llegada?), y estado del servidor.
    - Dado lo anterior, existen esencialmente dos tipos de *eventos*: llegada de clientes, y salida de clientes.
- En general, podríamos considerar eventos que no cambian las variables de estado de un sistema:
    - Un evento que marca el fin de la simulación.
    - Cambio en las reglas de operación del sistema.

# Mecanismos de avance de tiempo:

- Necesitamos conocer el tiempo *simulado*.
- Tiempo interno del sistema se llama *reloj de simulación*.

## Incremento Fijo

Se asume que los eventos ocurren en el conjunto $T_{sim} \in \{t_o, t_o + \Delta, t_o + 2\Delta, \ldots\}$.

## Avanzar al siguiente evento

Asume que los eventos pueden ocurrir en cualquier momento ($T_{sim} \in \mathbb{R}$), pero relacionados a un evento ($\| T_{sim} \| \leq \aleph_o$).

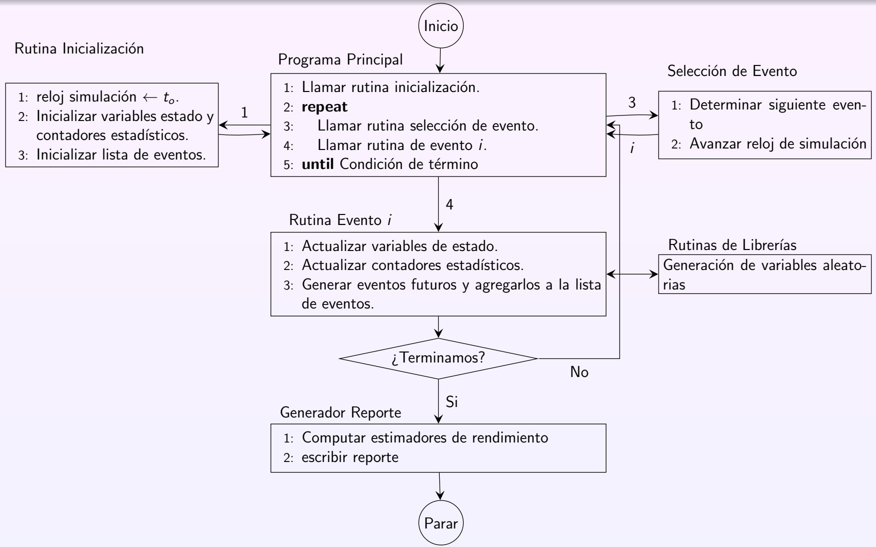
# Componentes y Organización

- Variables de Estado: Conjunto de variables de estado que describen el sistema en algún momento.
- Reloj de simulación: Variable que guarda el tiempo en el sistema.
- Lista de eventos: Lista (parcial) de eventos a realizarse en el futuro.
- Contadores Estadísticos: Variables que guardan los indicadores relevantes del sistema.
- Inicialización: Subrutina que inicializa modelo en $t_o$.
- Selección de evento: Subrutina que determina siguiente evento a realizarse.

# Componentes y Organización

- Rutinas de evento: Subrutina que actualiza el sistema cuando un evento ocurre.
- Rutinas Auxiliares: Generación de variables aleatorias, etc.
- Generador de Reporte: Subrutina que computa estimadores de las medidas de desempeño del sistema (basándose en los contadores estadísticos) cuando la simulación termina.
- Programa Principal: Programa que primero lee parámetros de entrada, inicializa el sistema, y después llama a la rutina de selección de eventos hasta el término de la simulación, y finalmente llama a la rutina de reporte.

# Flujo de un programa Típico de Simulación



# ¿Por que escribir software de simulación?

- Conocer como funcionan internamente todos los softwares de simulación ayuda a evitar errores conceptuales en su uso.
- En simulaciones complejas muchas veces es necesario programar algunas partes de la simulación para interactuar con softwares comerciales.
- Lenguajes de programación generales estan siempre disponibles, y son de bajo costo.
- Muchas simulaciones comerciales aun se programan en lenguajes generales.

# Generando Variables Aleatoreas:

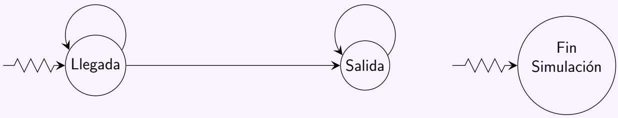
- En general asumiremos que existe un generador de numeros pseudo-aleatoreos $f()$ uniforme $(0, 1)$.
- Dada una distribuición acumulada $F(x) = \mathbb{P}(X \leq x)$, generaremos un valor aleatoreo usando la fórmula $x = F^{-1}(f())$
- Cuidado con funciones aleatoreas por defecto, en general de mala calidad.
- ¿Cuándo funciona?
    - Si $F$ es estrictamente creciente.
    - Con cuidado podemos hacerlo si $F$ es discreta, o si tiene *saltos*

# Simulación de un M/M/1

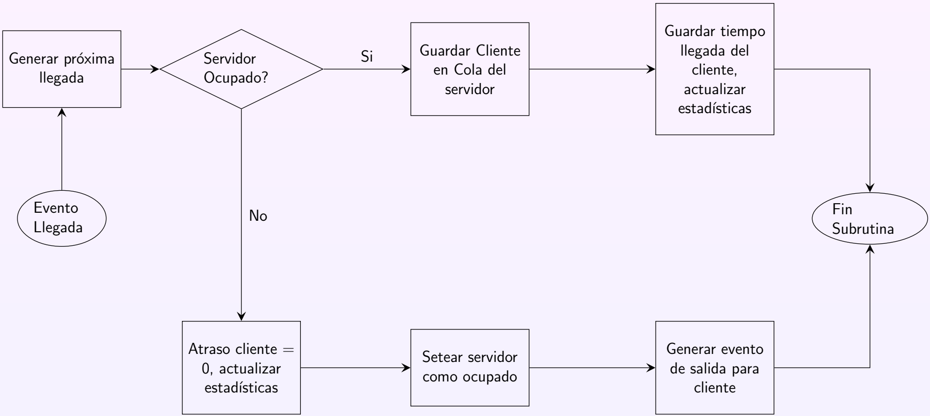
## Eventos

- Llegada de un Cliente $F(x) = 1 - e^{-\frac{x}{\beta_1}}$.
- Cliente Termina de ser atendido $F(x) = 1 - e^{-\frac{x}{\beta_2}}$.
- Cliente Comienza a ser atendido?

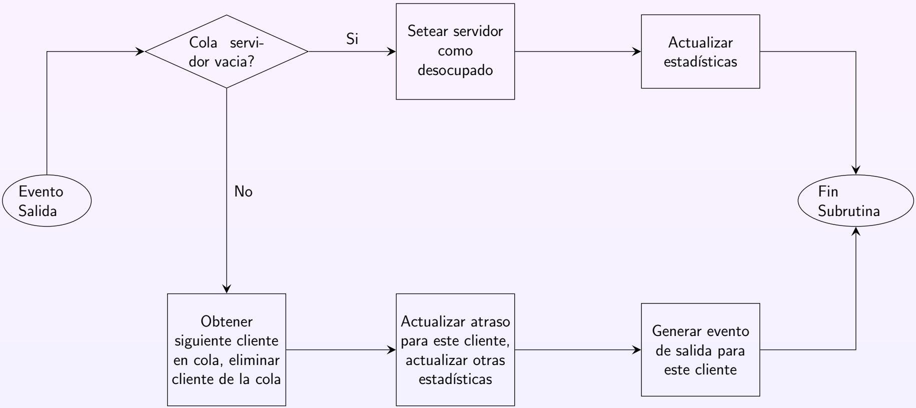
## Grafo de Eventos



# Lógica llegada de cliente



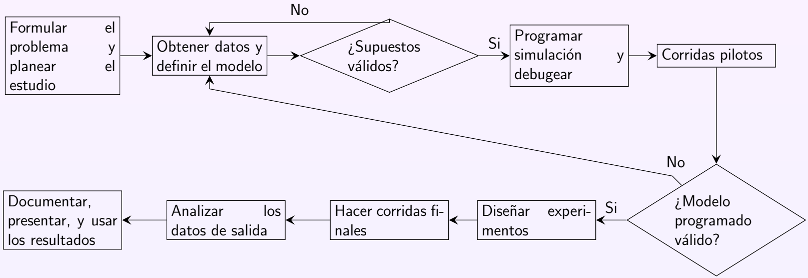
# Lógica salida de cliente



# Evaluando política de inventario

- Consideremos una política de inventario $(S, s)$ de revisión mensual, con costos por mantener items en bodega y back-orders.
- Demandas espaciadas *iid* $\approx$ exp(0,1), cantidades $\{1, 2, 3, 4\}$ con probabilidades $\{1/6, 1/3, 1/3, 1/6\}$.
- Cuando una orden se coloca, el tiempo de llegada es distribuido como $U[1/2, 1]$, y el costo es función de la cantidad demandada.

# Vista General



# Formular el problema y planear estudio

- Problema presentado por una unidad.
  - Problema puede estar mal definido, o en términos cualitativos.
  - Proceso Iterativo es necesario
- Varias reuniones con director proyecto, experto de simulación, experto en el sistema.
  - Definir objetivos del estudio.
  - Definir preguntas específicas a repsonder.
  - Definir medidas numércias para comparar diferentes configuraciones.
  - Alcance del modelo.
  - Diferentes configuraciones a probar.
  - Tiempo del estudio y recursos necesarios.
- Seleccionar el software a usar.

# Recoger Datos del Sistema

- Recoger información sobre estructura y reglas de operación del sistema.
  - Ningún individuo es suficiente.
  - Personas con conocimiento equivocado del sistema (buscar expertos reales).
  - Procedimientos pueden no estar formalizados.
- Recoger datos para definir parámetros del sistema y distribuciones de entrada.
- Formalizar información anterior en un documento de supuestos.
- Recoger datos sobre el desempeño del sistema real.

# Recoger Datos del Sistema

- Escoger nivel de detalle del sistema.
  - Objetivos del proyecto.
  - Métircas a utilizar.
  - Datos disponibles.
  - Problemas de credibilidad.
  - Opinion de los expertos del sistema.
  - Restricciones de tiempo/presupuesto.
- Empezar con un modelo simplificado y refinarlo a medida que sea necesario.
- Interactuar con el director del proyecto y expertos del sistema regularmente.

# ¿Son los Supuestos Válidos?

- Realizar una revision del documento de supuestos con los expertos del sistema y con el director del proyecto.
  - Asegurar que supuestos son correctos y completos.
  - Ayudar a la interacción entre miembros del equipo.
  - Incrementar el sentido de propiedad del modelo en el equipo.
  - Debería hacerce antes de comenzar a programar.

# Construir un programa y verificarlo

- Programar en un lenguaje general (C,C++,Java,C#) o en un software de simulación (Arena, Extend, Flexsim, ProModel).
  - Lenguajes generales tienen la ventaja de que usualmente uno es conocido.
  - Ofrecen control absoluto del programa.
  - Son mucho mas baratos de comprar (Licencias).
  - Pueden resultar en tiempos de ejecucion menores.
  - Lenguajes de simulacion resultan en menores tiempos de programación
  - Proveen de interfaces gráficas (Atractivas para gerencia).
- Debugear el programa.

# Corridas Piloto, ¿Es el modelo programado Válido?

- Ejecutar corridas pilotos del programa
- Comparar desempeño real del sistema con sistema simulado.
- Revisar consistencia de resultados con expertos del sistema real y con director proyecto.
- Realizar análisis de sensibilidad, identificando aspectos del sistema que necesitan mayor nivel de detalle o cuidado en el modelo.

# Diseño de Experimentos

- Para cada configuración de interes especificar:
  - Largo de cada corrida de simulación.
  - Período transiente de cada corrida (si necesario).
  - Número de simulaciones independientes a realizar, para así definir los correspondientes intervalos de confianza.

# Corridas Finales, Análisis de Resultados

- Ejecutar corridas principales.
- Objetivos principales en el análisis son:
  - Determinar desempeño absoluto de cada configuración analizada.
  - Comparar configuraciones alternativas en forma comparativa (análisis de tipo pareto).

# Documentar, presentar, y usar resultados

- Documentar supuestos utilizados.
- Documentar código del programa.
- Documentar creiterios de intervalos de confianza, etc.
- Presentar Resultados:
  - Uso de animaciones para presentar modelo a audiencia amplia.
  - Discutir proceso de validación de sistema.
  - Resultados se usarán en la medida de que sean validos y creibles.

# Supuestos Generales

- Asumimos que existe un generador de números aleatoreos uniforme.
- Nótese que estos no son continuos, tienen entre 32 o 56 bits de resolución.
- Diferencias menores a $10^{-9,6}$ o $10^{-16,8}$ no pueden observarse (recomendado: no menos de $10^{-6}$)
- Asumimos que aleatoredad del generador es buena.
- Asumimos que el generador es *eficiente*.

# Transformación Inversa

- Dado $U \sim U(0,1)$, $X \sim F$ probabilidad acumilada, retornar $X = F^{-1}(U)$.
- Ejemplo: $X \sim \exp(\beta)$ entonces
$$F(x) = \begin{cases} 1 - e^{-x/\beta} & x \geq 0 \\ 0 & \text{si no} \end{cases}$$
- $F^{-1}(u) = -\beta \log(1-u)$.
- Requiere generar un solo número aleatoreo.
- Extendible a $F$ no continuas, con saltos numerables.
- Más generalmente $X = \min\{x : F(x) \geq U\}$.
- Si $F^{-1}$ no se conoce, método numérico es necesario.
- Fácil producir distribuiciones truncadas.

# Compoción

- Aplicable a compoción convexa de variables aleatoreas.
- $F(x) = \sum_{i \in \mathbb{N}} p_i F_i(x)$, $p_i \geq 0$, $\sum_{i \in \mathbb{N}} p_i = 1$.
- Generar número $j$ tal que $P(j) = p_j$.
- Retornar $X$ con distribuición $F_j$.
- Ejemplo: $X \sim Trap(a)$, donde
  $$f(x) = \begin{cases} a + 2(1-a)x & 0 \leq x \leq 1 \\ 0 \text{si no} & \end{cases}.$$
- $f(x) = aI_{[0,1]}(x) + (1-a)2xI_{[0,1]}(x)$.
- Requiere sólo un números aleatoreo y generar sub-variable aleatorea.

# Convolución

- Aplicable a variables aleatoreas que son sumas de otras variables aleatoreas independientes.
- $X = \sum_{i=1}^{n} Y_i$.
- Generar $Y_i$ con la distribuición apropiada.
- Retornar $X = \sum_{i=1}^{n} Y_i$.
- Ejemplo: $X \sim$ m-Erlang($\beta$)
- $X = \sum_{i=1}^{m} \exp(\beta/m)$.

# Aceptar/Rechazar

- Tenemos $X$ con densidad $f(x)$ a soporte acotado $S$.
- Definimos $c = \max\{f(x) : x \in S\}$.
- Generar $x$ uniformemente en $S$.
- generar $Y$ uniformemente en $[0, c]$
- Retornar $x$ si $Y \leq f(x)$, si no, generar nuevamente $x, Y$.
- Útil cuando otros métodos son difícil de implementar.
- Dependiendo de $f$ puede requerir muchas generaciones de números aleatoreos.

# Definiciones

- $\mathbb{E}(X) = \int_{Dom(X)} x \cdot f(x) \cdot dx.$
- $\mathbb{V}\mathrm{ar}(X) = \mathbb{E}(X^2) - \mathbb{E}(X)^2 = \mathbb{E}((X - \mathbb{E}(X))^2).$
- $\mathbb{C}\mathrm{ov}(X, Y) = \mathbb{E}((X - \mathbb{E}(X))(Y - \mathbb{E}(Y))) = \mathbb{E}(XY) - \mathbb{E}(X)\mathbb{E}(Y).$
- Si $X$, $Y$ son independientes $\mathbb{C}\mathrm{ov}(X, Y) = 0$.
- La reciproca no es cierta.
- $\mathbb{C}\mathrm{or}(X, Y) = \dfrac{\mathbb{C}\mathrm{ov}(X, Y)}{\sqrt{\mathbb{V}\mathrm{ar}(X)\mathbb{V}\mathrm{ar}(Y)}}.$

# Estimadores

- Consideramos $X_i : i = 1, \ldots, n$ una muestra de $X$.
- $\overline{X}(n) = \dfrac{\sum_{i=1}^{n} X_i}{n}$
- $\overline{X}(n)$ es un estimador no sesgado de $\mathbb{E}(X)$.
- $S^2(n) = \dfrac{\sum_{i=1}^{n} (X_i - \overline{X}(n))^2}{n-1}$.
- $S^2(n)$ es un estimador no sesgado de $\mathbb{V}\mathrm{ar}(X)$.
- $\mathbb{V}\mathrm{ar}(\overline{X}(n)) = \dfrac{\mathbb{V}\mathrm{ar}(X)}{n}$.

# Teorema Central del Límite

- Consideramos $\{X_i\}_{i\in\mathbb{N}}$ variables aleatoreas iid con $\mathbb{E}(X_i) = \mu$ y $\mathbb{V}\mathrm{ar}(X_i) = \sigma^2$.
- Definimos $Z_i = \dfrac{\overline{X}(n) - \mu}{\sqrt{\frac{\sigma^2}{n}}}$.
- Llamamos $F_n(z) = \mathbb{P}(Z_n \leq x)$

## Teorema Central del Límite

$$\lim_{n\to\infty} F_n(z) = \Phi(z)$$

Donde $\Phi(z) = \dfrac{1}{\sqrt{2\pi}} \displaystyle\int_{-\infty}^{z} e^{-y^2/2} dy$.

# Teorema Central del Límite

- Básicamente el TCL dice que $Z_n \sim \mathcal{N}(0,1)$ cuando $n$ es grande.
- Otro problema es que $Z_n$ esta definido en términos de $\sigma^2$.
- Definimos $t_n = \dfrac{\overline{X}(n) - \mu}{\sqrt{\frac{S^2(n)}{n}}}$.
- Se puede demostrar que $t_n$ tambien converge a una $\mathcal{N}(0,1)$.
- De ahi podemos decir que $\mathbb{P}(-z_{1-\alpha/2} \leq t_n \leq z_{1-\alpha/2}) \approx 1 - \alpha$ para $n$ suficientemente grande.

# Intervalos de Confianza

- De lo anterior, podemos concluir que

$$\mathbb{P}(l(n) \leq \mu \leq u(n)) = 1 - \alpha$$

para $n$ suficientemente grande, donde

$$l(n) = \overline{X}(n) - z_{1-\alpha/2}\sqrt{\frac{S^2(n)}{n}}$$

y

$$u(n) = \overline{X}(n) + z_{1-\alpha/2}\sqrt{\frac{S^2(n)}{n}}$$

# Intervalos de Confianza

- ¿Qué pasa si $X_i \sim \mathcal{N}(0,1)$?
- En ese caso $t_n \sim$ T-student de $n-1$ grados de libertad.
- Intervalo de confianza exacto esta dado por $\overline{X}(n) \pm t_{n-1,1-\alpha/2}\sqrt{S^2(n)/n}$.
- Se tiene que estos intervalos son mayores a considerar que $t_n \sim \mathcal{N}(0,1)$.
- En general, debemos preguntarnos ¿Qué significa $n$ suficientemente grande?.

# Cobertura Real

- Consideramos intervalos de confianza derivados de la T-student.
- Distinto número de muestras $n = 5$, 10, 20, y 40.
- Consideramos $X_i$ iid con distintas distribuciones.
- Comparamos cobertura real del intervalo estimado a 90 % sobre 500 repeticiones.

| Dist | $\nu$ | n=5 | n=10 | n=20 | n=40 |
|---|---|---|---|---|---|
| Normal | 0.00 | 0.910 | 0.902 | 0.898 | 0.900 |
| Exponencial | 2.00 | 0.854 | 0.878 | 0.870 | 0.890 |
| Chi$^2$ | 2.83 | 0.810 | 0.830 | 0.848 | 0.890 |
| Lognormal | 6.18 | 0.758 | 0.768 | 0.842 | 0.852 |
| Hiper-exp | 6.43 | 0.584 | 0.586 | 0.682 | 0.774 |

# Midiendo Simetría de Distribuciones

- ¿Qué es $\nu$?
- $\nu = \dfrac{\mathbb{E}((X-\mu)^3)}{\sigma^3}$.
- $\nu$ es una medida de simetria de la distribución.
- Simetría de una distribuición es un factor importante al momento de determinar cuando *n es suficientemente grande* en el contexto del TCL.
- En general, No deberíamos mirar solamente a $\mu$, si no que también a $\sigma^2$ cuando describimos una distribuición.

# ¿Cómo obtenemos variables iid?

- Consideremos un sistema de simulación donde hay sólo una medida de desempeño, que es reportada en distintos puntos $J$ durante la simulación.
- Suponemos además que ejecutamos $n$ corridas independientes de la simulación, esto define $X_{i,j}$ con $i = 1, \ldots, n$ y $j \in J$.
- Asumiendo buenos números aleatoreaos, podemos considerar $\{X_{i,j}\}_{i=1}^{n}$ como variables iid.
- Desafortunadamente $\{X_{i,j}\}_{j \in J}$ en la práctica no son independientes, de hecho, usualmente, tienen correlación positiva.

# Algunos Ejemplos prácticos

- Consideramos un modelo M/M/1 con taza de ocupación $\rho = ,9$.
- Tratamos de estimar promedio de lso 25 primeros atrazos.
- Computamos 500 intervalos de confianza basados en 5,10,20 y 40 replicaciones.
- Comparamos proporción de intervalos *correctos* y su ancho medio.

# M/M/1, estimando $d_{25}$

| n | cobertura | intervalo 90 % | medio ancho |
|---|---|---|---|
| 5 | 0.880 | ±0.024 | 0.67 |
| 10 | 0.864 | ±0.025 | 0.44 |
| 20 | 0.886 | ±0.023 | 0.30 |
| 40 | 0.914 | ±0.021 | 0.21 |

# Tiempo medio a Falla

- Sistema con tres componentes.
- Sistema funciona mientras componente 1 funcione y componente 2 o componente 3 funcionen.
- Tiempo falla $G = \text{mín}\{G_1, \text{máx}\{G_2, G_3\}\}$, $G_i$ es Weibull(0.5,1).

| n | cobertura | intervalo 90 % | medio ancho |
|---|---|---|---|
| 5 | 0.708 | ±0.033 | 1.16 |
| 10 | 0.750 | ±0.032 | 0.82 |
| 20 | 0.800 | ±0.029 | 0.60 |
| 40 | 0.840 | ±0.027 | 0.44 |

# El Problema

- En general queremos comparar distintas configuraciones.
- Significa estimar parametros y compararlos.
- ¿Cuándo podemos decir que son distintos?

### Ejemplo

Compararemos un sistema M/M/1 con un sistema M/M/2. En sistema M/M/1 llegadas son 10 por unidad de tiempo, y atendemos 11 clientes por minutos. En sistema M/M/2 llegadas son 10 por unidad de tiempo, y cada servidor atiende 5.5 clientes por minutos.

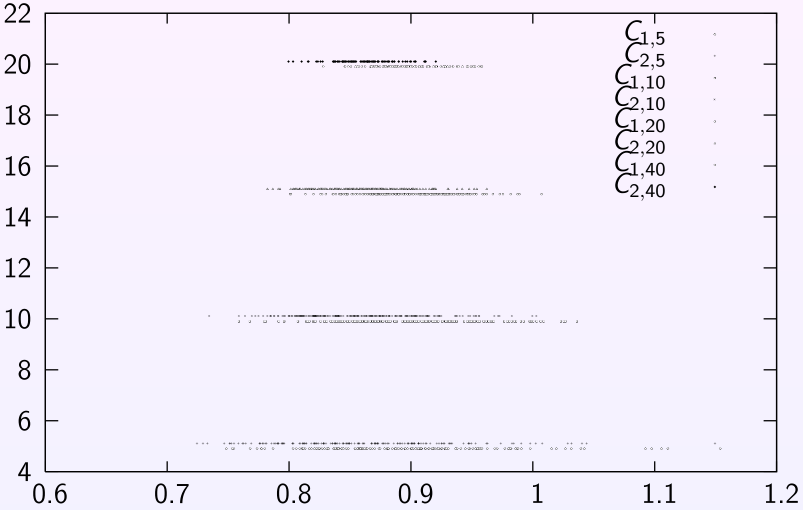
# Comparando estimadores de $\mu$

- Supongamos que ambas configuraciones cuestan lo mismo.
- Escoger sólo dependera de calidad de servicio.
- Medimos calidad de servicio como tiempo espera promedio.

## Primer reflejo

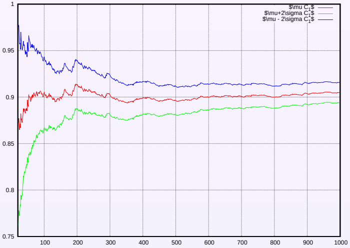
Supongamos que tenemos un *especialista* que sabe de simulación. Simula ambos sistemas, y computa una estimación de $\mu$ para ambos sistemas con $k$ corridas independientes.
El decide escoger el sistema con mejor tiempo de espera *estimada*

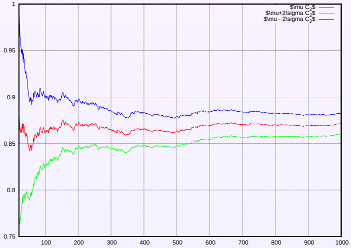
# ¿Cómo nos va?


22
20
18
16
14
12
10
8
6
4
0.6
0.7
0.8
0.9
1
1.1
1.2
$C_{1,5}$
$C_{2,5}$
$C_{1,10}$
$C_{2,10}$
$C_{1,20}$
$C_{2,20}$
$C_{1,40}$
$C_{2,40}$

# Otro Enfoque

- Un mejor enfoque seria compupar intervalos de confianza para ambos $\mu$.
- Habria que alanzar un nivel donde ambos intervalos no se overlapen.

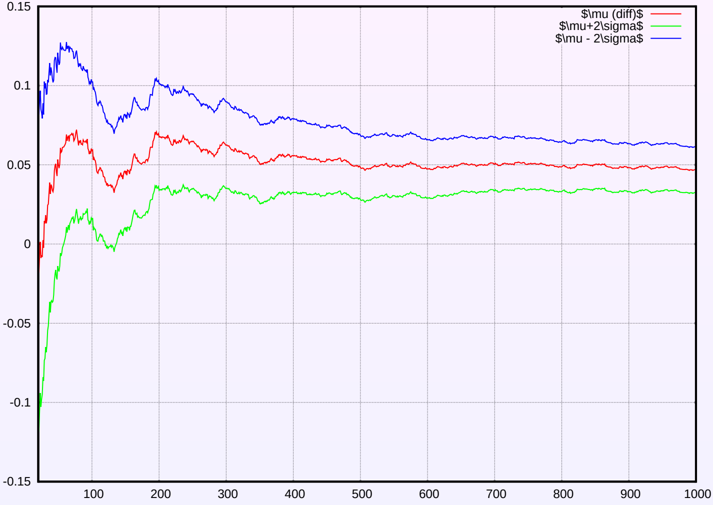


# Algunas Observaciones

- El enfoque anterior funciona.... pero
- Típicamente requiere un número de replicaciones altas.
- Sufre de los problemas de no-simetria de las distribuiciones subjacentes.
- El problema anterior podríamos solucionarlo.
- Considerando diferencia de los estimadores.
- Analizamos $Z_n = X_n - Y_n$.
- $Z_n$ tiende a tener una alta simetria.
- Decimos que configuraciones son distintas si 0 no esta en intervalo de confianza.

# ¿Cómo nos va con este enfoque?



# Podemos Mejorar?

- Otra forma de comparar sistemas o configuraciones.
- Exacerbar differencias en lo sistemas.
- Comparar bajo situaciones de stress del sistema.
- Ello conlleva a diferencias más sustanciales en los estadísticos.

# Caso de múltiples configuraciones:

## Comparación con Standard:

- Elegimos una configuración base $X^0$.
- Comparamos $k$ configuraciones $X^i : i = 1, \ldots, k$.
- Computamos intervalos de confianza de $X^0 - X^i : i = 1, \ldots, k$ a nivel $1 - \alpha/k$.
- Obtenemos Intervalo de confianza global de $1 - \alpha$.

# Caso de múltiples configuraciones:

## Comparacion todos los pares:

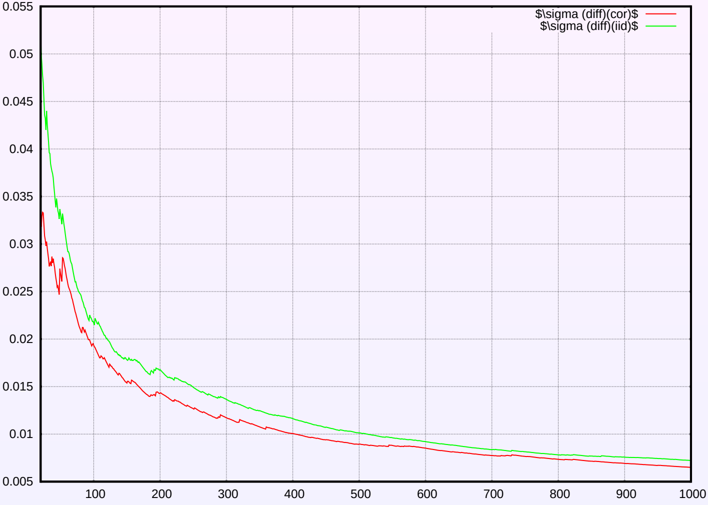
- Dado $k$ configuraciones $X^i : i = 1, \ldots, k$.
- Comparamos todos los pares $i, j$.
- Computamos intervalos de confianza para $X^i - X^j : i, j = 1, \ldots, k, i \neq j$ de nivel $1 - \alpha/k(k-1)$.
- Obtenemos Intervalo de confianza global de $1 - \alpha$.

# ¿Qué comparamos cuando comparamos?

- Típicamente comparamos un sistema bajo distintas reglas de operación.
- Los enfoques anteriores comparar ciegamente.
- Llevan a una alta varibilidad.
- ¿Qué queremos realmente?
- Comparar sistema en condiciones lo más cercanas posibles.
- En nuestro ejemplo, ¿qué significaría esto?
- Sistema bajo *misma* demanda.
- Notese que ahora correlación de $X$ e $Y$ no es cero.
- Lleva a menores varianzas.
- requiere menos replicaciones para resultados confiables.

# Impacto en Ejemplo

# Impacto en Ejemplo



# ¿Qué quedó afuera?

- Generar variables aleatoreas correlacionadas.
- Validando supuestos del modelo
  - Habría que hacer análisis estadísticos.
- Case de sistemas en estado estacionario.
- Análisis de sistemas oscilantes.
- Buscando buenos generadores de números aleatoreos.