FACULTAD DE CIENCIAS ECONÓMICAS Y DE ADMINISTRACIÓN

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

Microsimulación de Sistemas Sociales

Docente: Daniel Ciganda

1era Clase

20 de Agosto de 2025

Aspectos Generales Sobre el Curso

- Miércoles de 15 a 17hs Salón 004 Aulario
- 18/08 al 05/10 - 1er Período Clases
- 20/10 al 07/12 - 2ndo Período Clases
- Cada sesión consiste de aprox. 1 hora de exposición teórica + 1 hora de laboratorio en R
- Evaluación continua - Entrega periódicas de los ejercicios prácticos
- Dos trabajos escritos + defensa oral.
- Requisitos para la exoneración del examen:
  - Al menos 80 % de asistencia.
  - Entrega de 2/3 de los ejercicios prácticos.
  - Mínimo de 50 % del puntaje en cada uno de los proyectos.
- Requisitos para ganar el derecho a examen:
  - Al menos 50 % de asistencia.
  - Mínimo de 40 % del puntaje total de los dos proyectos, obtenidos con la entrega de uno de los proyectos o con la suma de ambos.

Agenda de la Sesión

Parte 1: Fundamentos Teóricos

1. Las dos lentes del modelado: Macro vs. Micro
2. Profundizando en el nivel micro:
   - Microsimulación Dinámica (MSD)
   - Modelos Basados en Agentes (MBA)
3. Un continuo de complejidad
4. El desafío computacional

Parte 2: Laboratorio Práctico Programación Eficiente en R

- Vectorización
- Paralelización
- Perfilamiento de código

Objetivos del curso

El propósito central de este curso es comprender **diferentes tradiciones de modelado a nivel micro**, con especial foco en la microsimulación dinámica (MSD) y en los modelos basados en agentes (MBA).

La clave para construir **buenos modelos** y entender la **dinámica de los sistemas sociales** reside en la **sintesis** de dos fuerzas:

- **Profundidad teórica** para capturar mecanismos y patrones de comportamiento (el énfasis de los MBA).
- **Rigor empírico** para anclar los modelos en datos observados y potenciar su impacto para la resolución de problemas concretos (el énfasis de la MSD).

En la integración de teoría y datos se encuentra la clave para construir buenos modelos.

El Nivel de Análisis: Macro vs. Micro

Modelado Macro

- **Enfoque:** Top-down.
- **Unidad:** Variables agregadas a nivel de sistema (PBI, tasa de desempleo, población total).
- **Pregunta Clave:** ¿Cúal es la dinámica y los patrones que caracterizan al sistema?
- **Ver el bosque**.

Modelado Micro

- **Enfoque:** Bottom-up.
- **Unidad:** Entidades individuales (personas, hogares, empresas).
- **Pregunta Clave:** ¿Cómo los comportamientos y las interacciones individuales generan resultados a nivel macro?
- **Ver los árboles**.

Idea Central

La elección del nivel no es solo técnica, sino que refleja supuestos teóricos sobre **dónde residen los mecanismos causales** de un fenómeno social.

Ventajas y Desventajas

Macro: Ventajas

- Parsimonia y elegancia teórica.
- Menor demanda computacional y de datos.
- Útil para análisis de políticas a nivel agregado.

Macro: Desventajas

- Falacia del “agente representativo”.
- Dificultad para modelar heterogeneidad.
- Los mecanismos causales pueden ser una “caja negra”.

Micro: Ventajas

- Captura explícita de la heterogeneidad de la población.
- Mecanismos causales a nivel individual son explícitos.
- Permite estudiar la emergencia de fenómenos macro (feedbacks).
- Análisis distributivo detallado (quién gana/pierde).

Micro: Desventajas

- Alta demanda de datos (microdatos longitudinales).
- Computacionalmente intensivo.
- Riesgo de sobre-parametrización y complejidad innecesaria.

De la Matemática a la Simulación

Modelado Macro

- Altamente **matematizable**.
- Se apoya en ecuaciones diferenciales, sistemas dinámicos, equilibrio general.
- Permite derivar soluciones analíticas o semi-analíticas.

Modelado Micro

- La heterogeneidad y las interacciones requieren un lenguaje **mas flexible**. Los resultados exactos dan lugar a las aproximaciones y al cálculo numérico
- Aquí entra la **simulación** como herramienta central.
- Tanto la **MSD** como los **MBA** se construyen como modelos computacionales.

Cuando bajamos al nivel micro, lo computacional aparece como el “lenguaje natural”.

El foco del curso: dos enfoques micro

Microsimulación Dinámica (MSD)

- Basada en **probabilidades de transición** estimadas de datos.
- Los individuos son *portadores de riesgos*: envejecen, nacen, mueren, cambian de estado.
- Usada extensamente en **política pública** (pensiones, impuestos, proyecciones demográficas).

Modelos Basados en Agentes (MBA)

- Basados en **reglas de decisión e interacción**.
- Los agentes son *actores autónomos*: perciben, deciden, aprenden.
- Usados para explorar **dinámicas emergentes** (mercados, redes, segregación).

Idea central

Ambos modelan individuos, pero:

- MSD = microdatos + probabilidades $\Rightarrow$ proyección de agregados.
- MBA = reglas + interacciones $\Rightarrow$ patrones emergentes.

MSD y MBA: orígenes, usos y herramientas

	Microsimulación Dinámica (MSD)	**Modelos Basados en Agentes (MBA / ABM)**
Origen	Economía aplicada y políticas públicas (años 60–80; *Orcutt, 1957*). Enraizada en microdatos de censos y encuestas.	Ciencia de la computación y sistemas complejos (autómatas celulares; *Schelling, 1971*).
Propósito central	Proyección y evaluación de políticas: Generación de escenarios e impactos distributivos.	Explicación de mecanismos y emergencia: “cómo” reglas locales generan patrones macro.
Unidades y datos	Individuos/hogares de encuestas representativas; fuerte anclaje empírico.	Agentes con reglas/heurísticas; puede ser teórico, experimental o empírico.
Validación	Ajuste a agregados observados.	Replicación de “hechos estilizados”.

Ejemplos Clásicos en Cada Tradición

Microsimulación Dinámica (MSD)

- **DYNACAN (Canadá)** Desarrollado para proyectar la sostenibilidad del sistema de pensiones. Modela nacimientos, muertes y trayectorias laborales.
- **CORSIM / DYNASIM (EE.UU.)** Modelos pioneros de los años 70–80 que simulan la vida de individuos y hogares (educación, empleo, matrimonio, fecundidad) para evaluar escenarios de política social y económica.

Modelos Basados en Agentes (MBA)

- **Schelling (1971) – Segregación residencial** Muestra cómo preferencias individuales mínimas sobre vecinos producen fuertes patrones de segregación urbana.
- **Epstein & Axtell (1996) – Sugarscape** Una sociedad artificial donde agentes recolectan y comercian recursos, ilustrando cómo reglas locales generan desigualdad, migración y cultura emergente.

Limitaciones de Cada Enfoque

Microsimulación Dinámica (MSD)

- Dependencia fuerte de microdatos representativos: difícil en contextos sin buenas encuestas, censos o datos administrativos.
- Tiende a tratar a los individuos como *portadores de riesgos*, con poco espacio para agencia o estrategias.
- Enfoque aplicado: puede perder riqueza teórica.

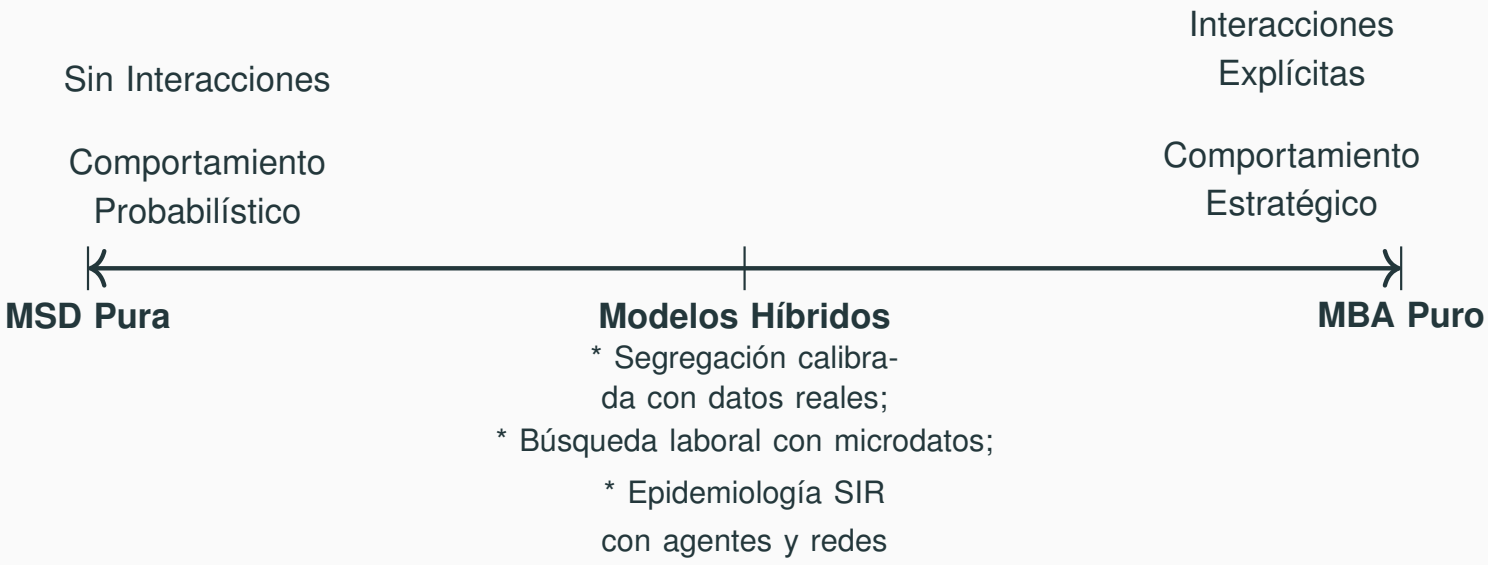
Modelos Basados en Agentes (MBA)

- Alta flexibilidad $\rightarrow$ riesgo de *"sandbox"*: cualquier cosa es posible, pero no siempre verificable.
- Validación empírica desafiante: difícil anclaje en datos reales.
- Complejidad computacional y parametrización excesiva.

Un Continuo, no una Dicotomía

Idea Central

En la práctica, los modelos más innovadores se sitúan en un punto intermedio, combinando la riqueza empírica de la MSD con la riqueza teórica de los MBA.



De la Teoría a la Práctica: El Desafío Computacional

El problema de la escala en la simulación

Los modelos de simulación social, buscan representar sistemas complejos:

- Cientos de miles o millones de agentes (personas, hogares).
- Interacciones que se repiten en miles de pasos de tiempo.
- Múltiples réplicas para obtener resultados estadísticamente robustos.
- Múltiples réplicas para estimar of validar los modelos.

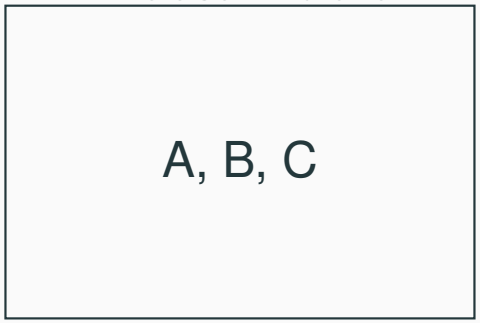
¿Por qué es crucial la eficiencia?

Un modelo ineficiente no es solo lento. Puede hacer que la **calibración**, el **análisis de sensibilidad** y la **exploración de escenarios** sean computacionalmente inviables.

Necesitamos herramientas para que nuestros modelos corran rápido

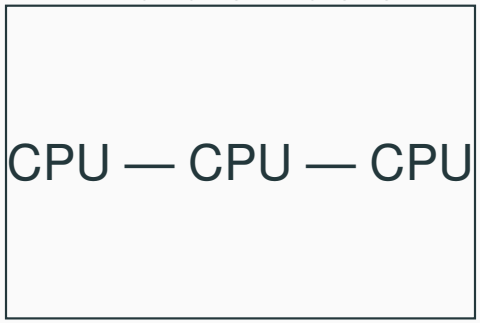
Herramientas para la Eficiencia en R

1. Vectorización



Reemplazar bucles (`for`) por operaciones que actúan sobre vectores completos.

2. Paralelización



Dividir tareas repetitivas (como correr múltiples simulaciones) para que se ejecuten simultáneamente en los distintos núcleos del procesador.

3. Perfilamiento

Usar herramientas de diagnóstico (`Rprof`) para encontrar los cuellos de botella (las partes más lentas de nuestro código).

Otras herramientas de optimización

Hoy vemos 3 palancas (vectorización, paralelización, perfilamiento), *pero hay más*: evitar copias, elegir buenas estructuras (p. ej. `data.table`), usar código compilado (Rcpp), y diseñar algoritmos con menor complejidad.

Código ineficiente (crece el vector)

```
set.seed(1)
n <- 20000
path <- numeric(0)   # ¡sin prealocar!
x <- 0
for (i in seq_len(n)) {
  step <- sample(c(-1, 1), 1)
  x <- x + step
  path <- c(path, x) # COPIA el vector
}
```

Código eficiente (prealoca)

```
set.seed(1)
n <- 20000
path <- numeric(n)   # prealocación
x <- 0
for (i in seq_len(n)) {
  step <- sample(c(-1, 1), 1)
  x <- x + step
  path[i] <- x       # asignación
}
```

¿Por qué es mala idea "hacer crecer" un vector en un bucle?

- En R, `c(path, x)` crea un *nuevo* vector y *copia* todo el contenido cada vez. Es como *comprar una caja más grande y mudar todos los libros* cada vez que se agrega uno. Prealocar es *comprar la caja del tamaño correcto* desde el principio.

¿Qué es Git?

- **Git** es un sistema de control de versiones.
- Git nos ayuda a generar un historial de los archivos de tu proyecto, permitiéndonos volver a versiones anteriores si es necesario.
- Permite que varias personas colaboren en un proyecto.

¿Qué es GitHub?

- **GitHub** es una plataforma web para alojar repositorios Git.
- Permite almacenar proyectos Git on line y trabajar con otras personas.
- Ofrece herramientas para el seguimiento de problemas, revisiones de código y más.

Comandos Básicos de Git

- `git init` - Inicializa un nuevo repositorio Git en el directorio actual, creando un subdirectorio oculto `.git` que contiene todos los metadatos del repositorio.
- `git clone <url-del-repositorio>` - Clona un repositorio remoto a tu máquina local.
- `git status` - Muestra el estado actual del repositorio, incluyendo archivos modificados, no rastreados, y cambios en el área de preparación (staging area).
- `git add <nombre-del-archivo>` - Añade los cambios de un archivo específico (o todos los archivos si se usa `.`) al área de preparación, preparándolos para ser confirmados en el siguiente commit.
- `git commit -m "mensaje"` - Graba los cambios del área de preparación en el historial del repositorio, creando un nuevo commit. El mensaje debe describir brevemente los cambios realizados.
- `git push` - Sube los commits locales al repositorio remoto asociado. Este comando envía el historial de commits y los objetos de Git al servidor remoto.
- `git pull` - Descarga y fusiona los cambios desde el repositorio remoto.

Cómo Descargar y Trabajar con los Materiales del Curso

- **1. Clonar el repositorio:**
  - Abrir Git Bash desde la locación deseada.
  - Ejecutar: `git clone https://github.com/dciganda/microsimulacion.git`
- **2. Crear un directorio de trabajo personal:**
  - Después de clonar, creamos un directorio separado fuera del repositorio donde copiamos los archivos que necesitamos para trabajar.
  - Nombre sugerido *"microsimulacion_apellido"*.
- **4. Trabajar en los archivos copiados:**
  - Para evitar conflictos en futuras actualizaciones del repositorio del curso.
- **5. Antes de cada clase:**
  - Desde Git bash ejecutamos `git pull` para actualizar el repositorio
  - Copiamos los archivos a *microsimulacion_apellido*.

IMPORTANTE: No realizar modificaciones a los archivos que están en el repositorio

La entrega de los ejercicios se realizará por correo a `daniel.ciganda@fcea.edu.uy`, recogiendo todos los laboratorios correspondientes a cada modulo en un archivo comprimido con nombre "laboratorios_moduloX_apellido"

Asunto del correo: *Entrega Laboratorios Modulo X*