CÁLCULO INFINITESIMAL con MAXIMA

Grado en Matemáticas

Renato Álvarez-Nodarse
Universidad de Sevilla

http://euler.us.es/~renato/clases.html

A modo de introducción

Los métodos del Cálculo diferencial son esenciales en la formación de cualquier persona interesada en una carrera científica o tecnológica y constituyen esencialmente la base de toda la ciencia moderna.

cálculo. (Del lat. calc**u**ulus).

1. m. Cómputo, cuenta o investigación que se hace de algo por medio de operaciones matemáticas.

A modo de introducción o justificación.

Hasta principios del siglo XX se cálculaba a mano ... lápiz, papel y el cerebro ...

A modo de introducción o justificación.

Hasta principios del siglo XX se cálculaba a mano ... lápiz, papel y el cerebro ...

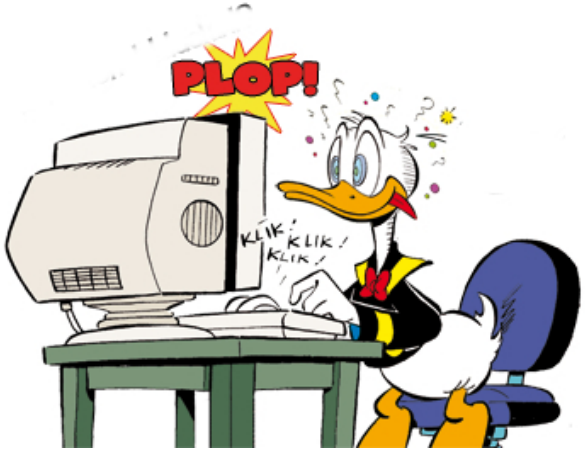
pero a finales de la 2ª Guerra Mundial ...

A modo de introducción o justificación.

Hasta principios del siglo XX se cálculaba a mano ... lápiz, papel y el cerebro ...

pero a finales de la 2ª Guerra Mundial ...

aparecen los ordenadores.



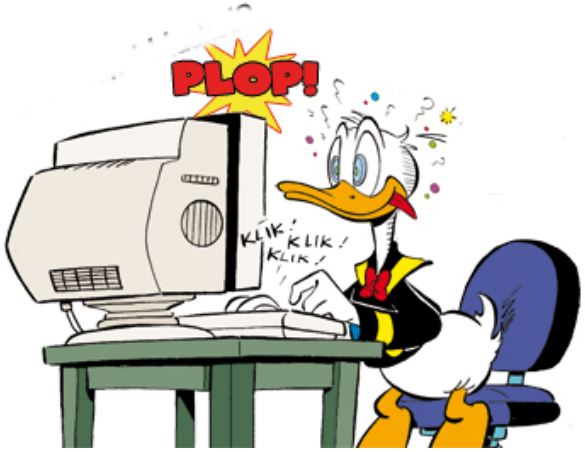
con lo que ello conlleva

A modo de introducción o justificación.

Hasta principios del siglo XX se cálculaba a mano ... lápiz, papel y el cerebro ...

pero a finales de la 2ª Guerra Mundial ...

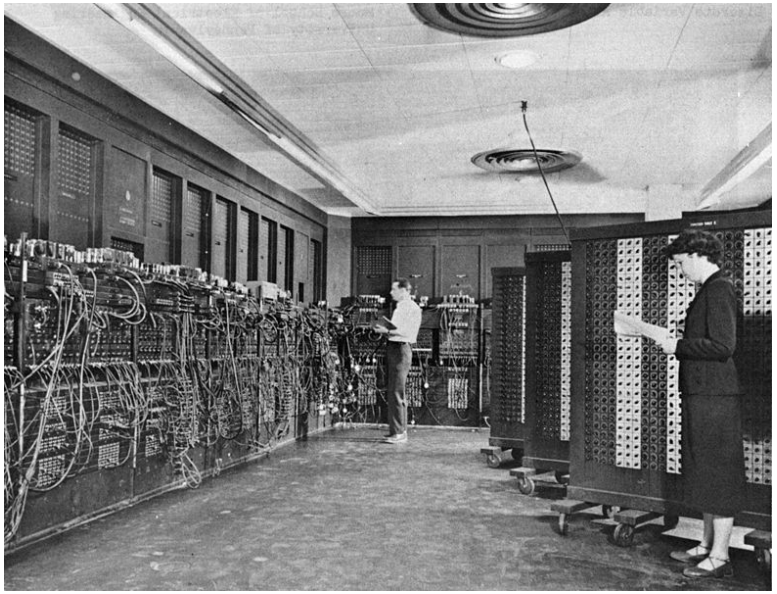
aparecen los ordenadores.



con lo que ello conlleva

Aparecen los ordenadores ... 194X

ENIAC – 1946



Precio 6 millones $
Ocupaba 167 m^2 y pesaba 27 000 Kg
ciclos básicos / seg: 200

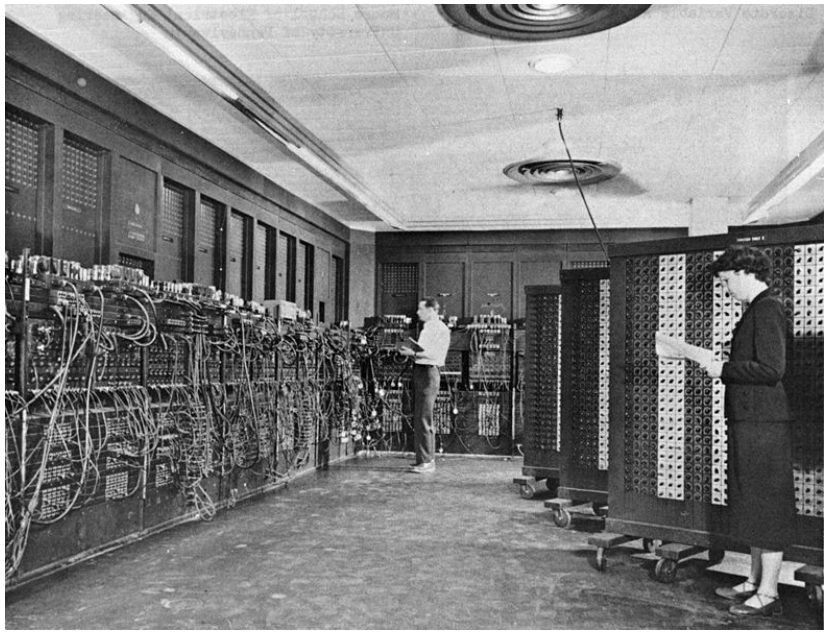
Toshiba z830 – 2012

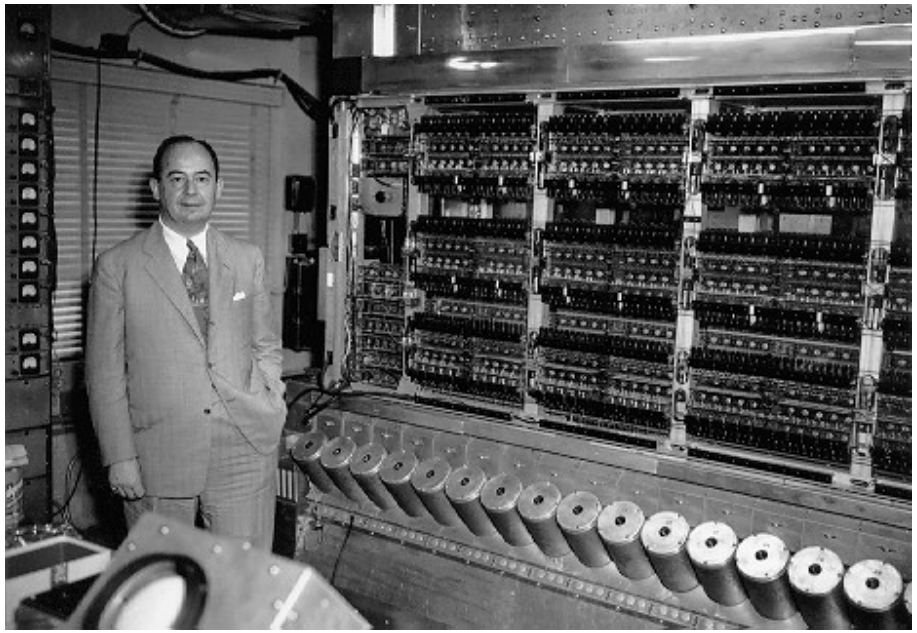
Precio 2000 $
Ocupa 0.07 m^2 y pesa 1.12 Kg
ciclos básicos / seg: $> 10^6$.

1CB= Buscar datos ⇒ Decodificar ⇒ Ejecutar ⇒ Guardar

Aparecen los ordenadores ... 194X

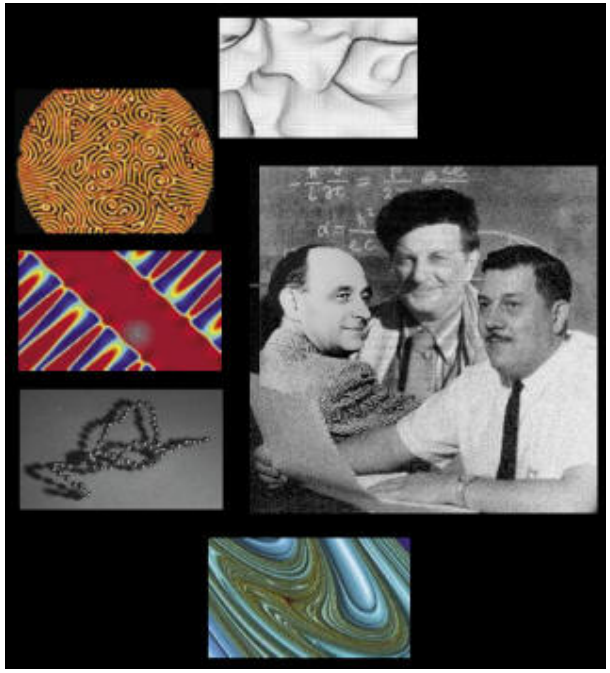
Los computadores actuales (hardware + software) fueron concebidos por John von Neumann el los años 194X (uso militar)





Antes de von Neuman cambiar el programa implicaba cambiar los circuitos eléctricos de la máquina.

Los ordenadores como instrumentos de experimentación matemática 1953



Fermi, Pasta y Ulam en el verano de 1953 usan el ordenador como un **instrumento de experimentación matemática**: simular un sistema físico concreto. La programadora fue Mary Tsingou.

El ordenador se llamaba MANIAC I (por sus siglas en inglés **M**athematical **A**nalyzer, **N**umerical **I**ntegrator, **a**nd **C**omputer).

¿Para qué usamos los ordenadores?

O sea, que aparte de los muchos usos lúdicos que puedan tener los ordenadores ...

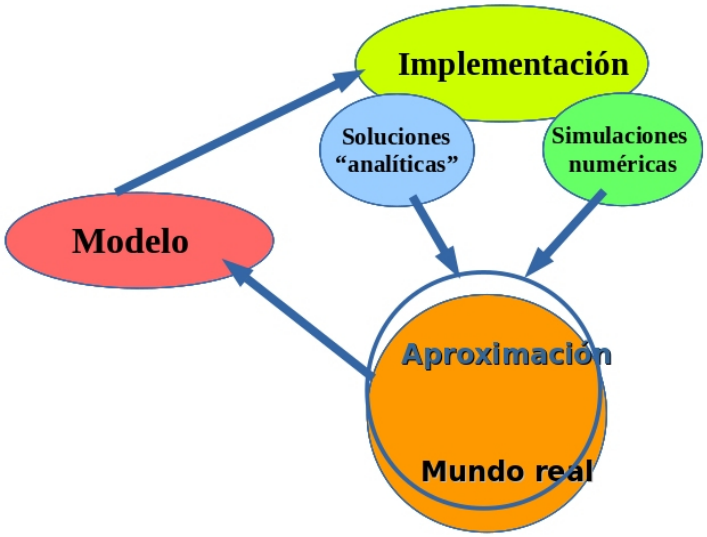
¿Para qué usamos los ordenadores?

O sea, que aparte de los muchos usos lúdicos que puedan tener los ordenadores ...

estos son una herramienta para resolver problemas y proyectos relacionados con las matemáticas y demás ciencias exactas.

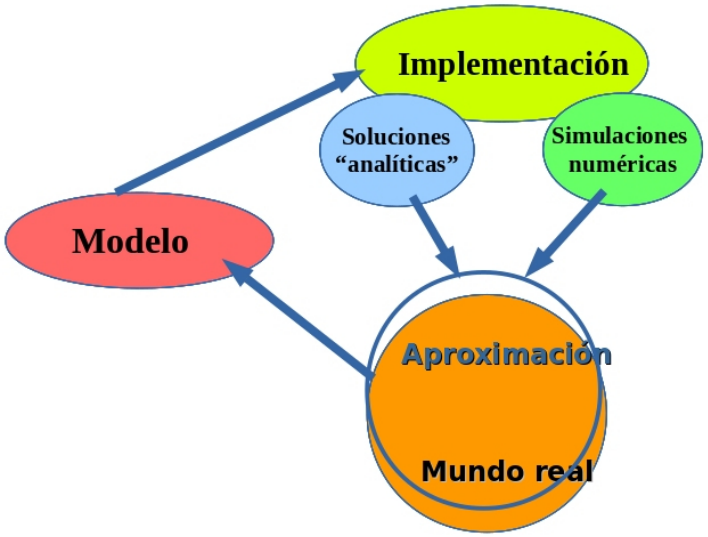
Por ejemplo: para *comprobar* las predicciones analíticas (teóricas) mediante experimentos numéricos: la **modelización**.

¿Qué es la modelización?

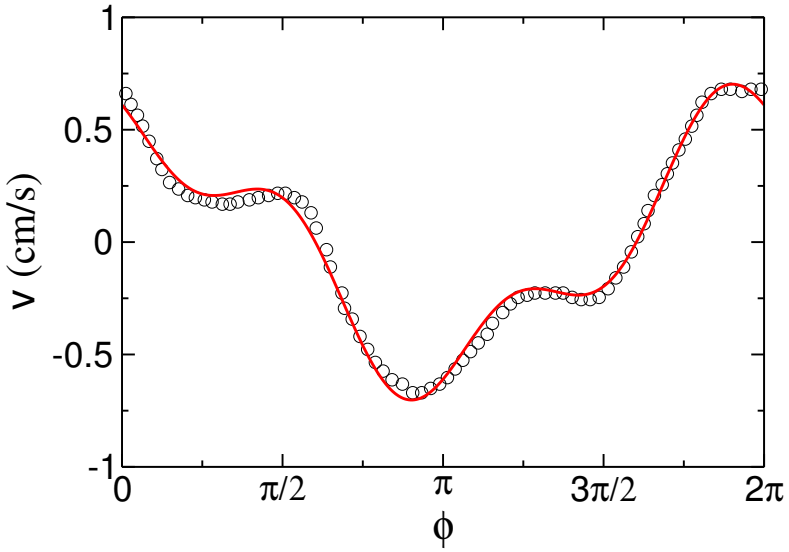


¿Cómo explicar y predecir los fenómenos naturales?

¿Qué es la modelización?



¿Cómo explicar y predecir los fenómenos naturales?



Movimiento de gotas de líquido sobre una placa en movimiento

El software: MAXIMA

Para trabajar con el ordenador necesitamos un **programa de cálculo matemático**. ¿¿ ??

El software: MAXIMA

Para trabajar con el ordenador necesitamos un **programa de cálculo matemático**. ¿¿ ??

Hemos elegido *software libre*.

El software: MAXIMA

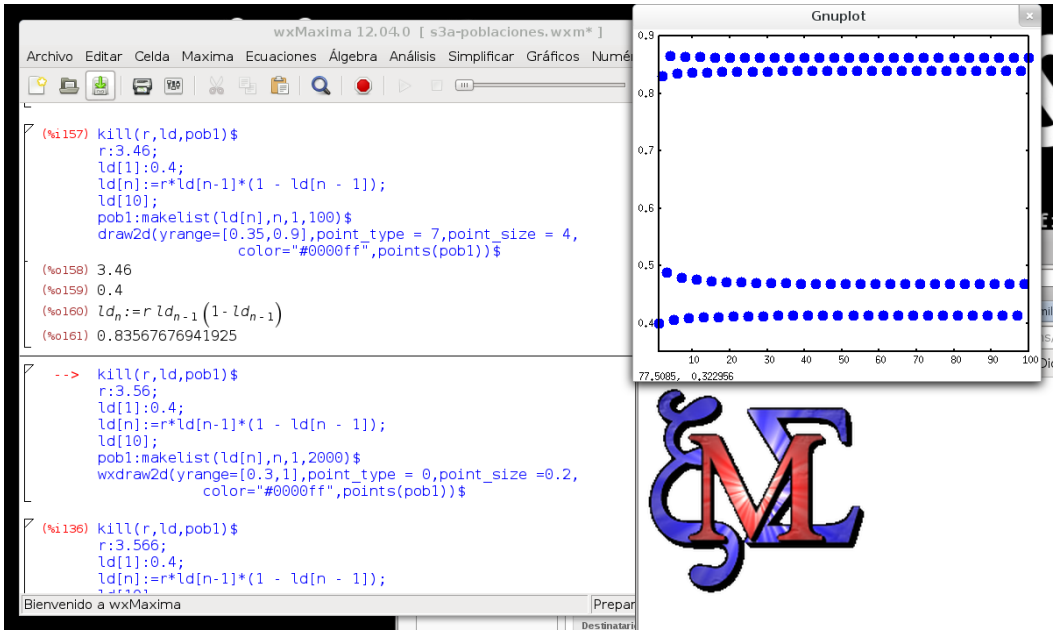
Para trabajar con el ordenador necesitamos un **programa de cálculo matemático**. ¿¿ ??

Hemos elegido *software libre*. **¿Por qué?**

El software: Maxima

Para trabajar con el ordenador necesitamos un **programa de cálculo matemático**. ¿¿ ??

Hemos elegido *software libre*. **¿Por qué? ¿Cuál?**



Maxima: programa simbólico/numérico con con licencia GNU/GPL accesibles por internet en

http://maxima.sourceforge.net

Como intértprete de MAXIMA usaremos WXMAXIMA

Buscar el icono y podemos comenzar

MAXIMA es un programa que funciona como una calculadora científica. Las operaciones aritméticas elementales son las habituales: $+$ suma, $-$ resta, $*$ multiplicación, / división, ^ potencias:

```
(%i1) 2+2; 3-3; 2*3; 5/10; 3^3;
(%o1) 4
(%o2) 0
(%o3) 6
(%o4) 1/2
(%o5) 27
```

MAXIMA es un programa que funciona como una calculadora científica. Las operaciones aritméticas elementales son las habituales: $+$ suma, $-$ resta, $*$ multiplicación, / división, ^ potencias:

```
(%i1) 2+2; 3-3; 2*3; 5/10; 3^3;
(%o1) 4
(%o2) 0
(%o3) 6
(%o4) 1/2
(%o5) 27
```

Dado que MAXIMA es un programa de cálculo *simbólico*, trabaja con variables definidas por letras:

```
(%i6) e;
(%o6) e
```

Las funciones (comandos) tienen el argumento (o los argumentos) que van entre paréntesis, como por ejemplo el comando `float(x)` que nos da como resultado el valor numérico de la variable x.

```
(%i7) float(e);
(%o7) e
(%i8) pi;
(%o8) pi
(%i9) float(pi);
(%o9) pi
```

por lo que su respuesta es la propia variable.
Para los valores numéricos de las constantes e o π MAXIMA usa

```
(%i10) %e;
(%o10) %e
(%i11) float(%e);
(%o11) 2.718281828459045
```

Por defecto MAXIMA trabaja en doble precisión, i.e., con 16 dígitos. Dado que es un programa simbólico, podemos definirle con cuantas cifras queremos trabajar con la variable `fpprec`. Para asignar valores a las variables dadas se usan los "dos puntos". Por ejemplo `e:%e`

```
(%i12) fpprec:50;
(%o12) 50
```

y lo comprobamos pidiendo a MAXIMA que nos de los valores de e y π con 50 cifras, para lo cual hay que usar el comando `bfloat`

```
(%i13) bfloat(%e); bfloat(%pi);
(%o13) 2.71828182845904523536028747135266249775724709370 7b0
(%o14) 3.14159265358979323846264338327950288419716939937 51b0
```

Ejercicio: Calcular e^7 numéricamente:

Por defecto MAXIMA trabaja en doble precisión, i.e., con 16 dígitos. Dado que es un programa simbólico, podemos definirle con cuantas cifras queremos trabajar con la variable `fpprec`. Para asignar valores a las variables dadas se usan los "dos puntos". Por ejemplo `e:%e`

```
(%i12) fpprec:50;
(%o12) 50
```

y lo comprobamos pidiendo a MAXIMA que nos de los valores de e y π con 50 cifras, para lo cual hay que usar el comando `bfloat`

```
(%i13) bfloat(%e); bfloat(%pi);
(%o13) 2.71828182845904523536028747135266249775724709370 7b0
(%o14) 3.14159265358979323846264338327950288419716939937 51b0
```

Ejercicio: Calcular e^7 numéricamente:

```
(%i15) float(%e^7);
(%i16) bfloat(%e^7);
(%i17) %e^7;
```

El orden de realización de las operaciones es el habitual. Así en la expresión

```
(%i18) (2+3^2)^3*(5+2^2);
(%o18) 11979
```

primero calcula las potencias dentro de cada paréntesis, luego las sumas, luego las potencias externas y finalmente la multiplicación.

El orden de realización de las operaciones es el habitual. Así en la expresión

```
(%i18) (2+3^2)^3*(5+2^2);
(%o18) 11979
```

primero calcula las potencias dentro de cada paréntesis, luego las sumas, luego las potencias externas y finalmente la multiplicación.

Para definir los valores de las variables se usan los dos puntos.

```
(%i19) x:123; y:321; x*y; x/y; x-y;
```

```
(%i26) z=2;
(%i27) z;
(%i28) x;
(%i29) y;
(%i30) x*y;
(%i31) x+z;
```

También es posible definir funciones. Hay múltiples formas en función de lo queramos hacer.

```
(%i32) f(x):= x^2 -x + 1;
(%i33) f(%pi);
```

Nótese que a no ser que pidamos a MAXIMA que trabaje numéricamente, sigue usando cálculo simbólico

```
(%i35) float(f(%pi));
```

También es posible definir funciones. Hay múltiples formas en función de lo queramos hacer.

```
(%i32) f(x):= x^2 -x + 1;
(%i33) f(%pi);
```

Nótese que a no ser que pidamos a MAXIMA que trabaje numéricamente, sigue usando cálculo simbólico

```
(%i35) float(f(%pi));
```

Otro detalle interesante a tener en cuenta es que MAXIMA contiene una ayuda completa que puede ser invocada desde la línea de comandos. Para ello escribimos `??` delante del comando *desconocido*

```
(%i36) ??float;
```

Si no existe ningún comando con ese nombre la respuesta es `false`

```
(%i37) ??renato;
```

Otra de las opciones de MAXIMA es su potencia gráfica. Para hacer las gráficas MAXIMA usa GNUPLOT, un potente paquete gráfico GNU. El comando más sencillo de usar es `plot2d`.

```
(%i39) wxplot2d([f(x)], [x,-5,5]);
Bad range: [123,-5,5].
Range must be of the form [variable,min,max]
 -- an error. To debug this try: debugmode(true);
```

Otra de las opciones de MAXIMA es su potencia gráfica. Para hacer las gráficas MAXIMA usa GNUPLOT, un potente paquete gráfico GNU. El comando más sencillo de usar es `plot2d`.

```
(%i39) wxplot2d([f(x)], [x,-5,5]);
Bad range: [123,-5,5].
Range must be of the form [variable,min,max]
 -- an error. To debug this try: debugmode(true);
```

¿Qué pasa?

Otra de las opciones de MAXIMA es su potencia gráfica. Para hacer las gráficas MAXIMA usa GNUPLOT, un potente paquete gráfico GNU. El comando más sencillo de usar es `plot2d`.

```
(%i39) wxplot2d([f(x)], [x,-5,5]);
Bad range: [123,-5,5].
Range must be of the form [variable,min,max]
 -- an error. To debug this try: debugmode(true);
```

¿Qué pasa?

Solución: Cambiar la variable por una no usada (por ejemplo `xx` o, usar una variable *muda*, es decir una sólo se use dentro del propio comando `wxplot`. Estas variables se definen colocando delante de la misma el signo `'`, por ejemplo, en vez de `x` usamos `'x`

```
(%i40) wxplot2d([f('x)], ['x,-5,5])$
```

Podemos representar varias funciones

```
(%i41) plot2d([f(xx),2*sin(xx),atan(xx)], [xx,-2,2])$
```

MAXIMA también dispone de una gran cantidad de funciones *elementales*, **exponencial**, **logaritmo**, **trigonométricas**, etc. Además las trata de forma simbólica.

```
(%i42) log(10);
(%i43) float(%);
(%i44) log(%e);
```

También podemos calcular factoriales

```
(%i45) 15!;
(%o45) 1307674368000
(%i46) factor(%);
```

Podemos representar varias funciones

```
(%i41) plot2d([f(xx),2*sin(xx),atan(xx)], [xx,-2,2])$
```

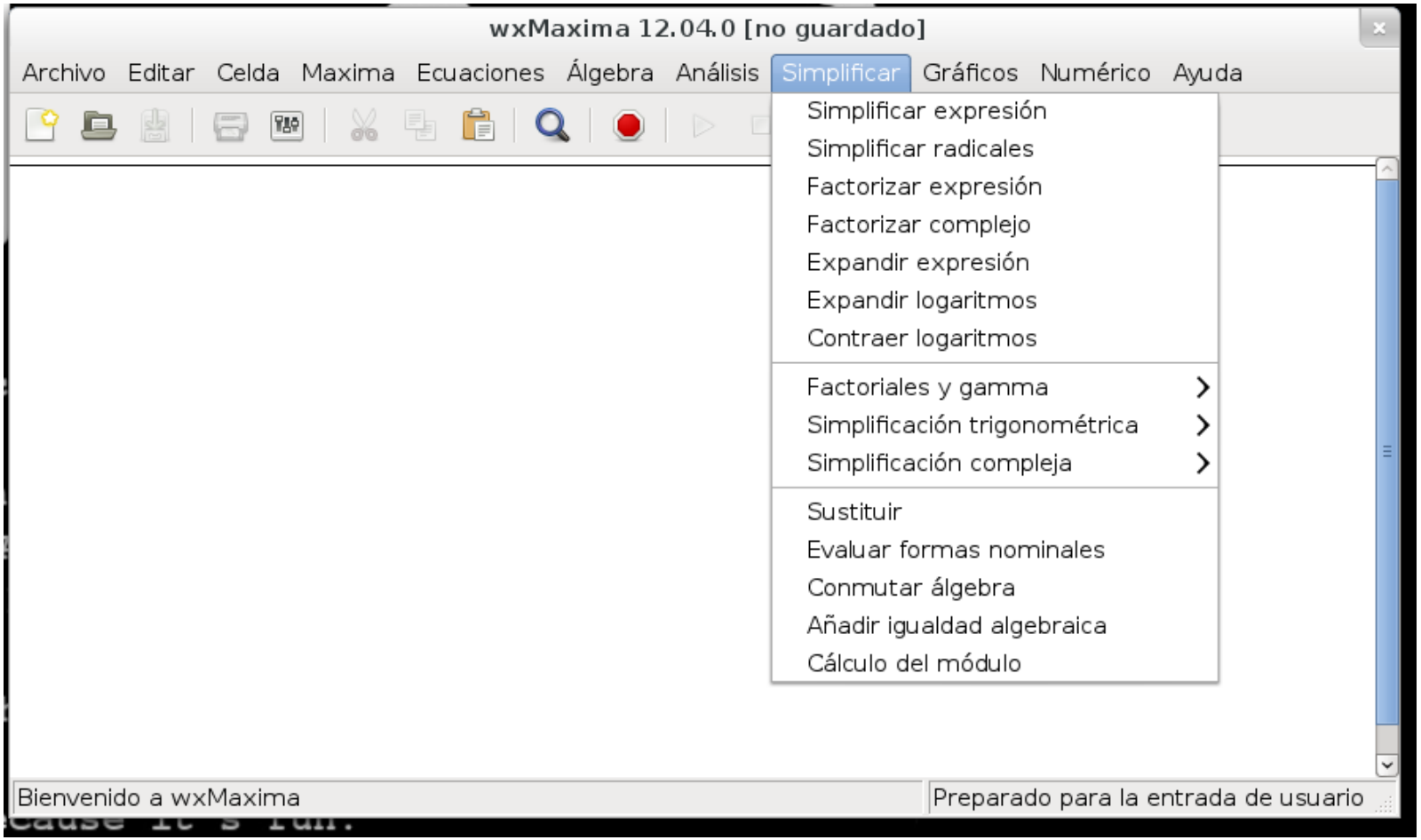
MAXIMA también dispone de una gran cantidad de funciones *elementales*, **exponencial**, **logaritmo**, **trigonométricas**, etc. Además las trata de forma simbólica.

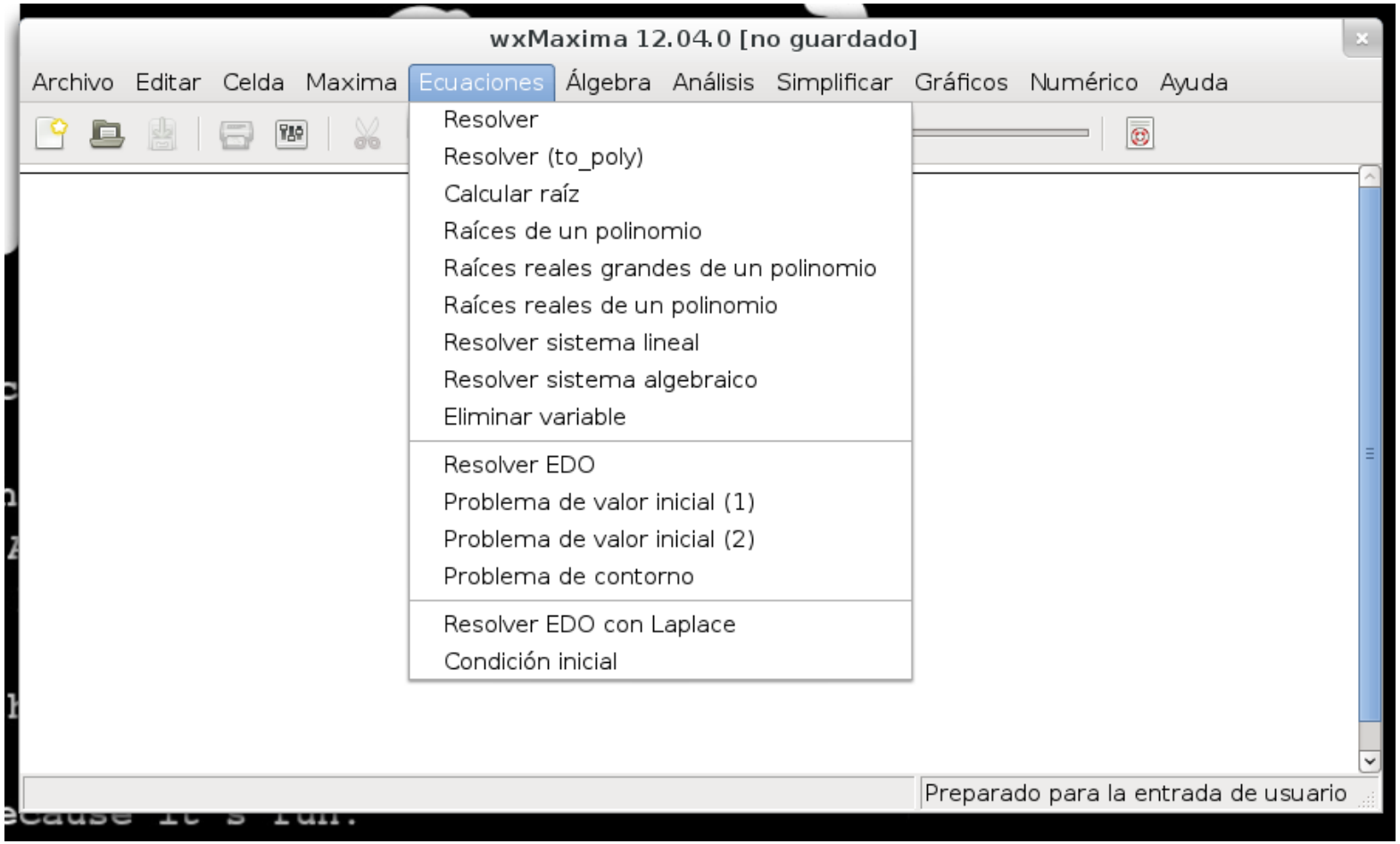
```
(%i42) log(10);
(%i43) float(%);
(%i44) log(%e);
```

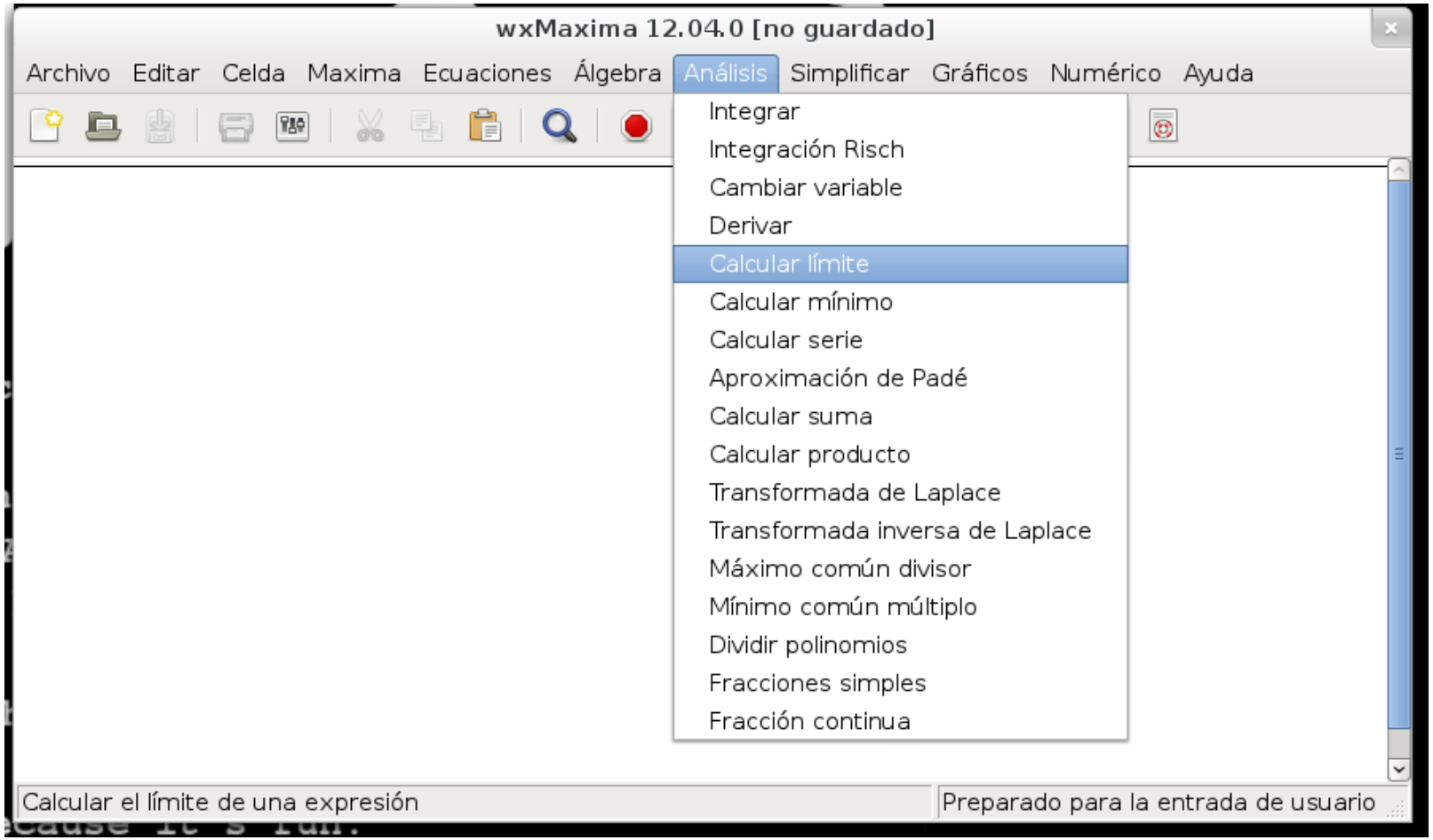
También podemos calcular factoriales

```
(%i45) 15!;
(%o45) 1307674368000
(%i46) factor(%);
```

¿Qué más podemos hacer?


wxMaxima 12.04.0 [no guardado]
Archivo
Editar
Celda
Maxima
Ecuaciones
Álgebra
Análisis
Simplificar
Gráficos
Numérico
Ayuda
Simplificar expresión
Simplificar radicales
Factorizar expresión
Factorizar complejo
Expandir expresión
Expandir logaritmos
Contraer logaritmos
Factoriales y gamma
Simplificación trigonométrica
Simplificación compleja
Sustituir
Evaluar formas nominales
Conmutar álgebra
Añadir igualdad algebraica
Cálculo del módulo
Bienvenido a wxMaxima
Preparado para la entrada de usuario


wxMaxima 12.04.0 [no guardado]
Archivo
Editar
Celda
Maxima
Ecuaciones
Álgebra
Análisis
Simplificar
Gráficos
Numérico
Ayuda
Resolver
Resolver (to_poly)
Calcular raíz
Raíces de un polinomio
Raíces reales grandes de un polinomio
Raíces reales de un polinomio
Resolver sistema lineal
Resolver sistema algebraico
Eliminar variable
Resolver EDO
Problema de valor inicial (1)
Problema de valor inicial (2)
Problema de contorno
Resolver EDO con Laplace
Condición inicial
Preparado para la entrada de usuario


wxMaxima 12.04.0 [no guardado]
Archivo
Editar
Celda
Maxima
Ecuaciones
Álgebra
Análisis
Simplificar
Gráficos
Numérico
Ayuda
Integrar
Integración Risch
Cambiar variable
Derivar
Calcular límite
Calcular mínimo
Calcular serie
Aproximación de Padé
Calcular suma
Calcular producto
Transformada de Laplace
Transformada inversa de Laplace
Máximo común divisor
Mínimo común múltiplo
Dividir polinomios
Fracciones simples
Fracción continua
Calcular el límite de una expresión
Preparado para la entrada de usuario

¡Y un sinfín de cosas más!

Vamos a ver algunas de ellas a lo largo de las próximas sesiones.

Para más detalles se recomienda visitar en la web de la asignatura el apartado *Software Libre de Matemáticas*

http://euler.us.es/~renato/clases.html#maxima

Cálculo diferencial con Maxima

Resolviendo ecuaciones:

```
(%i47) solve(x^2-a^2-4,x);
```

También se puede resolver respecto a la variable *a* (hazlo como ejercicio).

Resolviendo ecuaciones:

```
(%i47) solve(x^2-a^2-4,x);
```

También se puede resolver respecto a la variable a (hazlo como ejercicio).

```
(%i48) solve(x^2=-1);
(%i49) solve(x^5=1,x);
```

```
(%i50) rectform(%);
```

Ejercicio: Resuelve la ecuación $x^n = 1$ para $n \in \mathbb{N}$ cualquiera.

Resolviendo ecuaciones:

```
(%i47) solve(x^2-a^2-4,x);
```

También se puede resolver respecto a la variable a (hazlo como ejercicio).

```
(%i48) solve(x^2=-1);
(%i49) solve(x^5=1,x);
```

```
(%i50) rectform(%);
```

Ejercicio: Resuelve la ecuación $x^n = 1$ para $n \in \mathbb{N}$ cualquiera.

```
(%i51) solve(x^n=1,x);
"Is "n" an "integer"?"  y;
(%o51) [x=1]
```

Pero ¿y las demás soluciones?

Resolviendo ecuaciones:

```
(%i47) solve(x^2-a^2-4,x);
```

También se puede resolver respecto a la variable a (hazlo como ejercicio).

```
(%i48) solve(x^2=-1);
(%i49) solve(x^5=1,x);
```

```
(%i50) rectform(%);
```

Ejercicio: Resuelve la ecuación $x^n = 1$ para $n \in \mathbb{N}$ cualquiera.

```
(%i51) solve(x^n=1,x);
"Is "n" an "integer"?"  y;
(%o51) [x=1]
```

Pero ¿y las demás soluciones? Intenta "adivinarlas" a partir de los casos $n = 2, 3, 4, 5, 6, 7$ (usa `polarform(z)` si es necesario).

Calculando límites:

```
(%i52) limit(sin(a*x)/x,x,0);
```

Cuidado con los infinitos:

```
(%i53) limit(exp(1/(1-x)),x,1);
(%o53) und
(%i54) limit(exp(1/(1-x)),x,1,plus);
(%o54) 0
(%i55) limit(exp(1/(1-x)),x,1,minus);
(%o55) inf
```

Prueba con $\lim_{x\to 0} \log(x)$.

Calculando derivadas e integrales: el comando para calcular derivadas es `diff(f(x),x,k)` donde `f(x)` es la función a la que le vamos a calcular la derivada `k`-ésima respecto a la variable `x`:

```
(%i56) diff(sin(x^2+2),x);
(%o56) 2*x*cos(x^2+2)
(%i57) diff(x^x, x,3);
(%o57) x^(x-1)*(log(x)+(x-1)/x)+x^x*(log(x)+1)^3
           +2*x^(x-1)*(log(x)+1)
(%i58) diff(sin(x)^x, x);
(%o58) sin(x)^x*(log(sin(x))+(x*cos(x))/sin(x))
```

Calculando derivadas e integrales: el comando para calcular derivadas es `diff(f(x),x,k)` donde `f(x)` es la función a la que le vamos a calcular la derivada `k`-ésima respecto a la variable `x`:

```
(%i56) diff(sin(x^2+2),x);
(%o56) 2*x*cos(x^2+2)
(%i57) diff(x^x, x,3);
(%o57) x^(x-1)*(log(x)+(x-1)/x)+x^x*(log(x)+1)^3
           +2*x^(x-1)*(log(x)+1)
(%i58) diff(sin(x)^x, x);
(%o58) sin(x)^x*(log(sin(x))+(x*cos(x))/sin(x))
```

El comando `integrate(f(x),x)` calcula una primitiva de la función `f(x)`

```
(%i59) integrate(sin(2*x), x);
(%o59) -cos(2*x)/2
```

En MAXIMA se pueden hacer suposiciones sobre las variables. Por ejemplo calculemos la integral

```
(%i60) integrate (x^a*exp(-x), x, 0, inf);
Is  a+1  positive, negative, or zero? p;
(%o60) gamma(a+1)
```

este nos pregunta si la *a* es positiva, negativa, etc. Al responder nos da la solución (¿qué significa gamma?)
Si sabemos que, por ejemplo, la *a* es positiva, etc. podemos usar el comando assume que indica que al calcular la integral la *a* es un número mayor que uno.

```
(%i61) assume (a > 1)$
       integrate (x^a*exp(-x), x, 0, inf);
(%o61) gamma(a+1)
```

MAXIMA también dibuja gráficas 3D con el comando `plot3d` cuya sintaxis es

```
plot3d([fun1,fun2,...],[var1,ini,fin],[var2,ini,fin],...)
```

```
(%i62) kill(f,x,y)$
       f(x,y):= sin(x) + cos(y);
       wxplot3d(f(x,y), [x,-5,5], [y,-5,5])$
```

MAXIMA también dibuja gráficas 3D con el comando `plot3d` cuya sintaxis es

```
plot3d([fun1,fun2,...],[var1,ini,fin],[var2,ini,fin],...)
```

```
(%i62) kill(f,x,y)$
       f(x,y):= sin(x) + cos(y);
       wxplot3d(f(x,y), [x,-5,5], [y,-5,5])$
```

Para que MAXIMA reconozca el directorio local de trabajo (lo que es conveniente para importar y exportar ficheros) es conveniente definir la variable `file_search_maxima`.

La forma más sencilla de definirlo es mediante la opción "Añadir a ruta" en la pestaña "Maxima" del menú del programa.

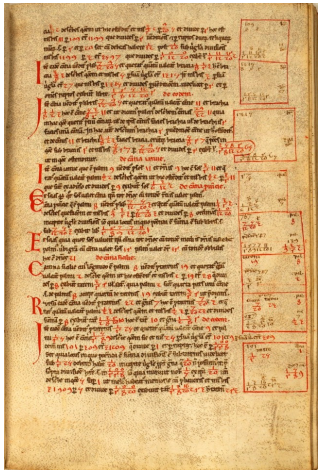
Sucesiones por recurrencia

$$x_1 = 0, \quad x_{n+1} = \sqrt{2 + x_n}$$

$$0,\ \sqrt{2},\ \sqrt{\sqrt{2}+2},\ \sqrt{\sqrt{\sqrt{2}+2}+2},\ \ldots$$

Motivando las sucesiones por recurrencia: Fibonacci y sus conejos

En 1202 apareció en Europa el tratado de aritmética *Liber Abaci* escrito por Leonardo de Pisa, mas conocido como Fibonacci. En este libro, uno de los primeros donde se usó el sistema de numeración indo-árabe en Europa, Fibonnaci propone el siguiente problema:

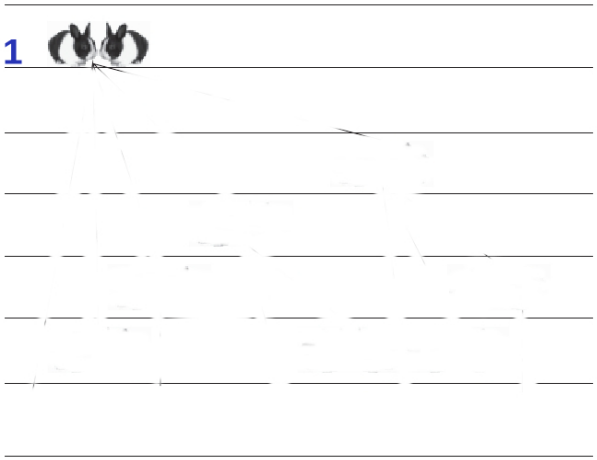


Cierto hombre tenía una pareja de conejos juntos en un lugar cerrado y uno desea saber cuántos son creados a partir de este par en un año cuando es su naturaleza parir otro par en un simple mes, y en el segundo mes los nacidos parir también.

O sea, ¿cuántos conejos hay después de n meses?

El modelo de poblaciones de Fibonacci 1202

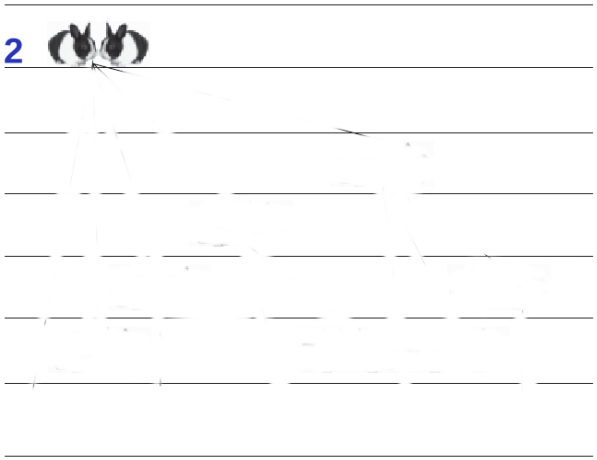
¿Cómo crece una población de conejos?



- Comenzamos con una única pareja de conejos

El modelo de poblaciones de Fibonacci 1202

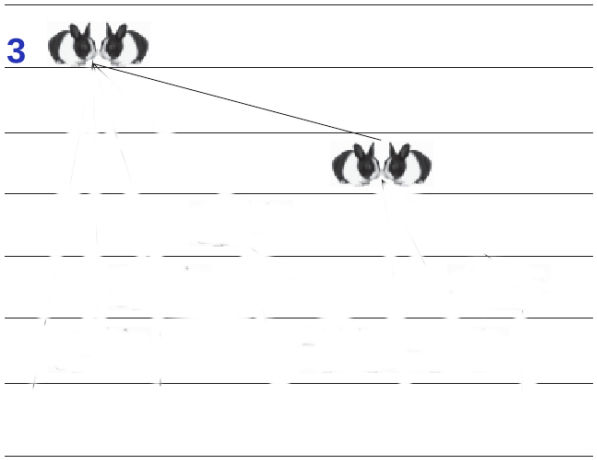
¿Cómo crece una población de conejos?



- Comenzamos con una única pareja de conejos
- Cada pareja de conejos *madura* pasado cierto tiempo T (un mes)

El modelo de poblaciones de Fibonacci 1202

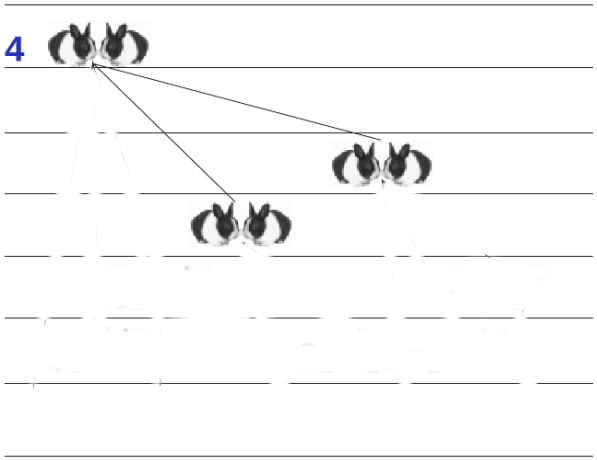
¿Cómo crece una población de conejos?



- Comenzamos con una única pareja de conejos
- Cada pareja de conejos *madura* pasado cierto tiempo T (un mes)
- Cada pareja madura de conejos *produce* una única nueva pareja de conejos cada temporada de crianza

El modelo de poblaciones de Fibonacci 1202

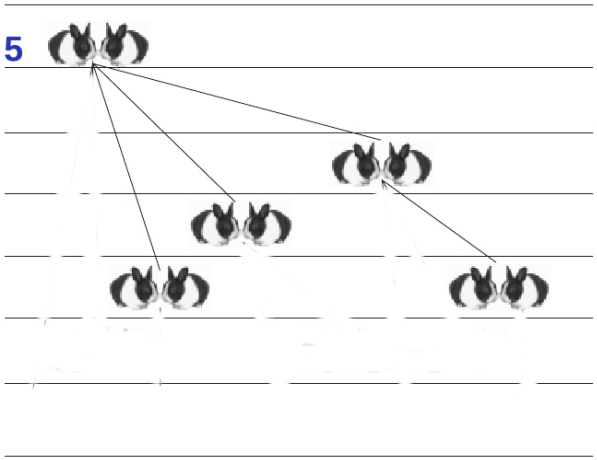
¿Cómo crece una población de conejos?



- Comenzamos con una única pareja de conejos
- Cada pareja de conejos *madura* pasado cierto tiempo T (un mes)
- Cada pareja madura de conejos *produce* una única nueva pareja de conejos cada temporada de crianza

El modelo de poblaciones de Fibonacci 1202

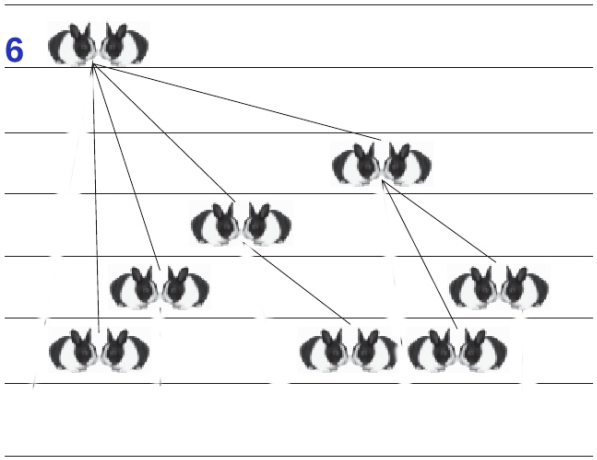
¿Cómo crece una población de conejos?



- Comenzamos con una única pareja de conejos
- Cada pareja de conejos *madura* pasado cierto tiempo T (un mes)
- Cada pareja madura de conejos *produce* una única nueva pareja de conejos cada temporada de crianza

El modelo de poblaciones de Fibonacci 1202

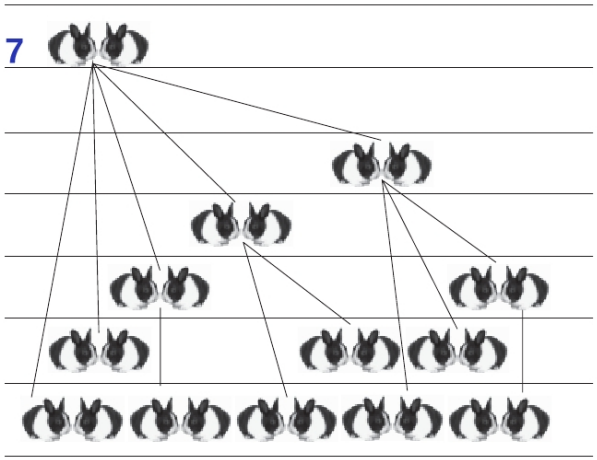
¿Cómo crece una población de conejos?



- Comenzamos con una única pareja de conejos
- Cada pareja de conejos *madura* pasado cierto tiempo T (un mes)
- Cada pareja madura de conejos *produce* una única nueva pareja de conejos cada temporada de crianza
- Los conejos son inmortales.

El modelo de poblaciones de Fibonacci 1202

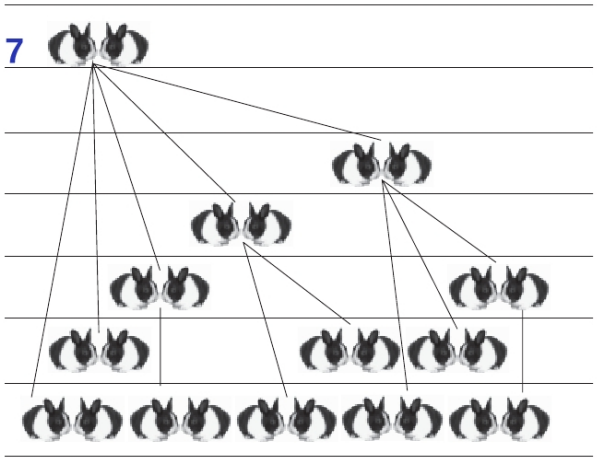
¿Cómo crece una población de conejos?



- Comenzamos con una única pareja de conejos
- Cada pareja de conejos *madura* pasado cierto tiempo T (un mes)
- Cada pareja madura de conejos *produce* una única nueva pareja de conejos cada temporada de crianza
- Los conejos son inmortales.

El modelo de poblaciones de Fibonacci 1202

¿Cómo crece una población de conejos?

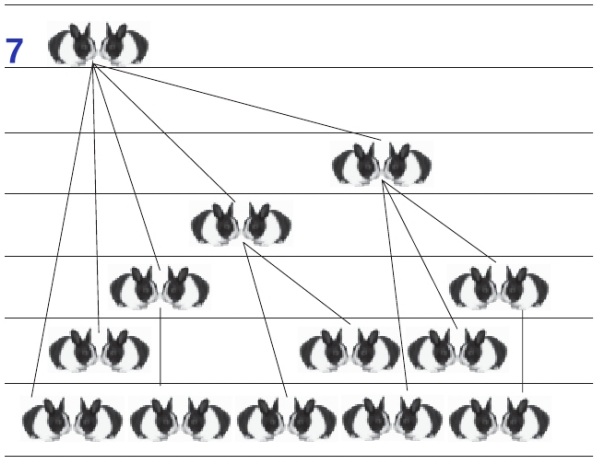


- Comenzamos con una única pareja de conejos
- Cada pareja de conejos *madura* pasado cierto tiempo T (un mes)
- Cada pareja madura de conejos *produce* una única nueva pareja de conejos cada temporada de crianza
- Los conejos son inmortales.

MODELIZACIÓN: PROBLEMA "*REAL*" + CONJUNTO DE REGLAS

El modelo de poblaciones de Fibonacci 1202

¿Cómo crece una población de conejos?



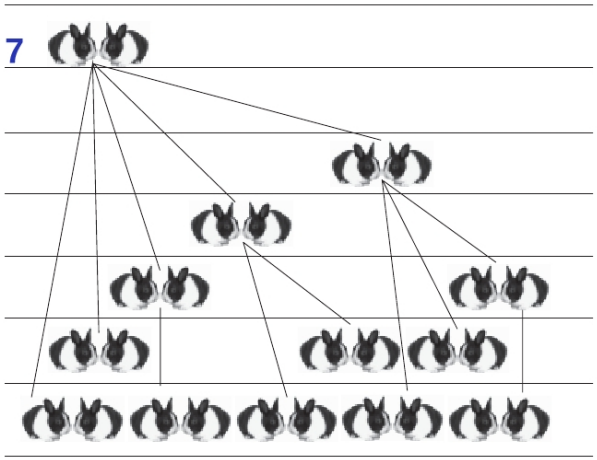
- Comenzamos con una única pareja de conejos
- Cada pareja de conejos *madura* pasado cierto tiempo T (un mes)
- Cada pareja madura de conejos *produce* una única nueva pareja de conejos cada temporada de crianza
- Los conejos son inmortales.

MODELIZACIÓN: PROBLEMA "*REAL*" + CONJUNTO DE REGLAS

$\Rightarrow$ ECUACIÓN MATEMÁTICA

El modelo de poblaciones de Fibonacci 1202

¿Cómo crece una población de conejos?

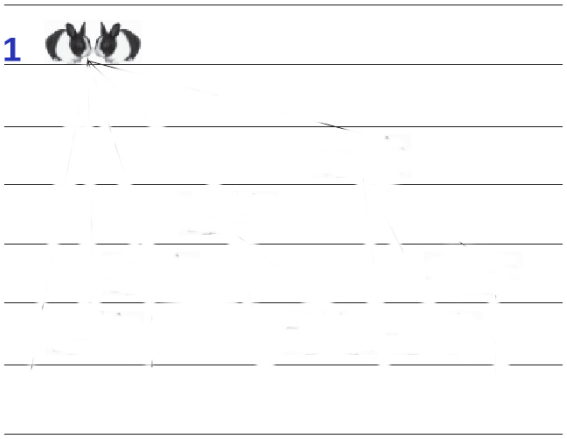


- Comenzamos con una única pareja de conejos
- Cada pareja de conejos *madura* pasado cierto tiempo T (un mes)
- Cada pareja madura de conejos *produce* una única nueva pareja de conejos cada temporada de crianza
- Los conejos son inmortales.

MODELIZACIÓN: PROBLEMA "*REAL*" + CONJUNTO DE REGLAS

$\boxed{\Rightarrow \text{ECUACIÓN MATEMÁTICA}}$ $\Rightarrow$ PREDICCIÓN

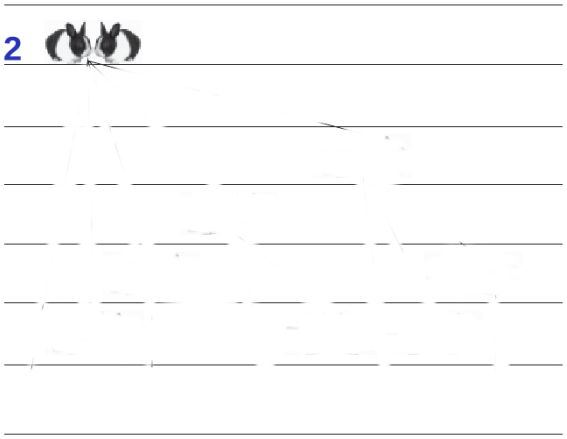
Modelos de poblaciones: Fibonacci 1202



Sea N_t el No. de parejas en cada temporada t: $\Longrightarrow N_{t+1} = N_t + N_{t-1}$, $t = 1, 2, 3, \ldots$.

Si $N_1 = N_2 = 1$ la fórmula anterior genera la famosa sucesión de Fibonacci:

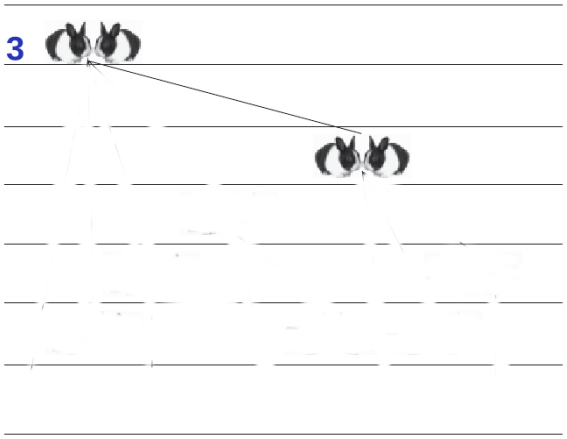
Modelos de poblaciones: Fibonacci 1202



Sea N_t el No. de parejas en cada temporada t: $\Longrightarrow N_{t+1} = N_t + N_{t-1}$, $t = 1, 2, 3, \ldots$.
Si $N_1 = N_2 = 1$ la fórmula anterior genera la famosa sucesión de Fibonacci:

$$1, 1,$$

Modelos de poblaciones: Fibonacci 1202

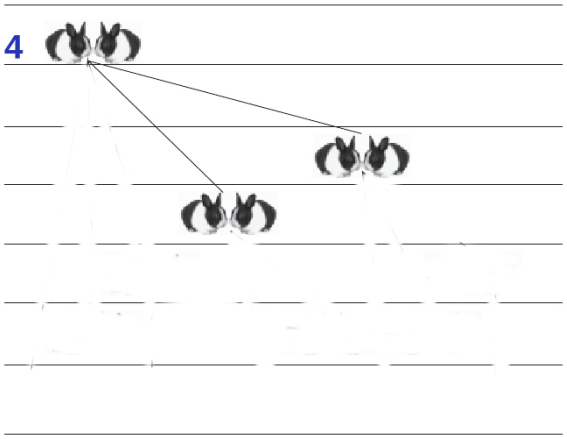


Sea N_t el No. de parejas en cada temporada t: $\Longrightarrow N_{t+1} = N_t + N_{t-1}$, $t = 1, 2, 3, \ldots$.

Si $N_1 = N_2 = 1$ la fórmula anterior genera la famosa sucesión de Fibonacci:

$$1, 1, 2,$$

Modelos de poblaciones: Fibonacci 1202

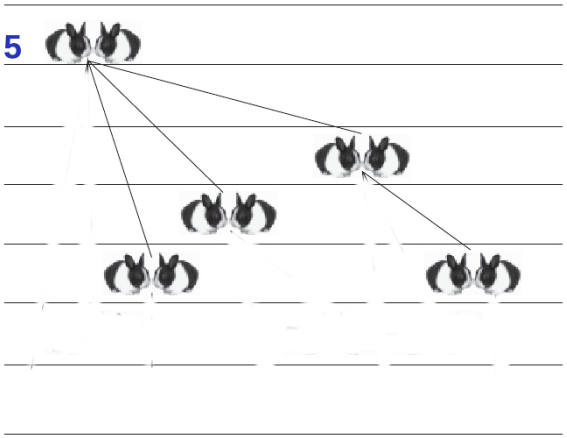


Sea N_t el No. de parejas en cada temporada t: $\Longrightarrow N_{t+1} = N_t + N_{t-1}$, $t = 1, 2, 3, \ldots$.

Si $N_1 = N_2 = 1$ la fórmula anterior genera la famosa sucesión de Fibonacci:

$$1, 1, 2, 3,$$

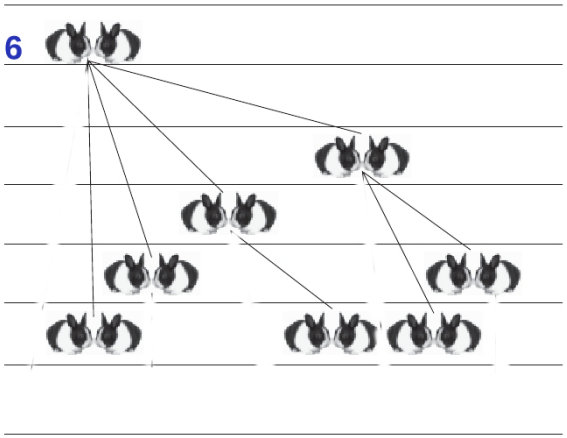
Modelos de poblaciones: Fibonacci 1202



Sea N_t el No. de parejas en cada temporada t: $\Longrightarrow N_{t+1} = N_t + N_{t-1}$, $t = 1, 2, 3, \ldots$.
Si $N_1 = N_2 = 1$ la fórmula anterior genera la famosa sucesión de Fibonacci:

$$1, 1, 2, 3, 5,$$

Modelos de poblaciones: Fibonacci 1202

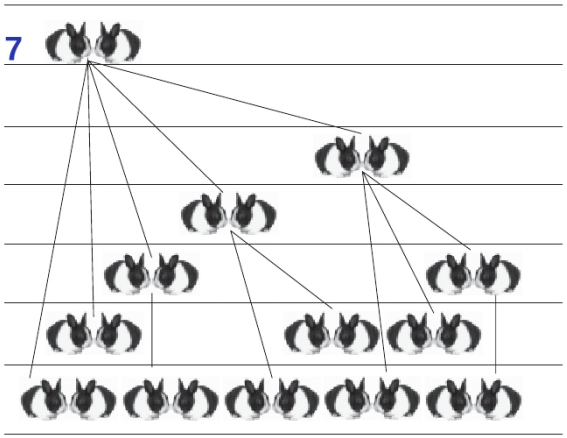


Sea N_t el No. de parejas en cada temporada t: $\Longrightarrow N_{t+1} = N_t + N_{t-1}$, $t = 1, 2, 3, \ldots$.

Si $N_1 = N_2 = 1$ la fórmula anterior genera la famosa sucesión de Fibonacci:

$$1, 1, 2, 3, 5, 8,$$

Modelos de poblaciones: Fibonacci 1202

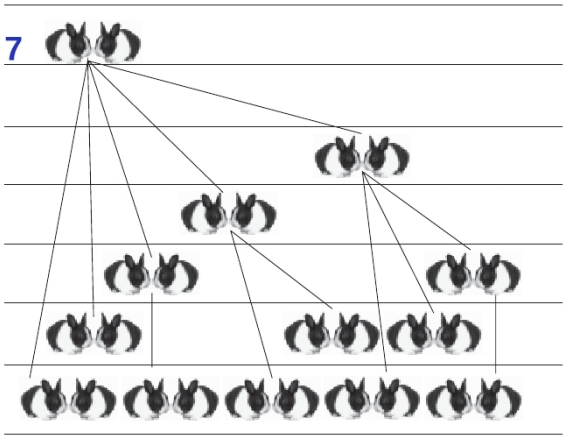


Sea N_t el No. de parejas en cada temporada t: $\Longrightarrow N_{t+1} = N_t + N_{t-1}$, $t = 1, 2, 3, \ldots$.

Si $N_1 = N_2 = 1$ la fórmula anterior genera la famosa sucesión de Fibonacci:

$$1, 1, 2, 3, 5, 8, 13,$$

Modelos de poblaciones: Fibonacci 1202

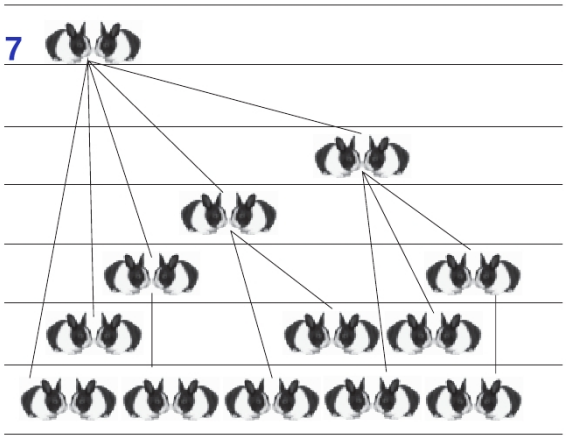


Sea N_t el No. de parejas en cada temporada t: $\Longrightarrow$ $N_{t+1} = N_t + N_{t-1}$, $t = 1, 2, 3, \ldots$.

Si $N_1 = N_2 = 1$ la fórmula anterior genera la famosa sucesión de Fibonacci:

$$1, 1, 2, 3, 5, 8, 13, 21, \ldots \quad ¿\ N_t\ ? \quad \Longrightarrow$$

Modelos de poblaciones: Fibonacci 1202



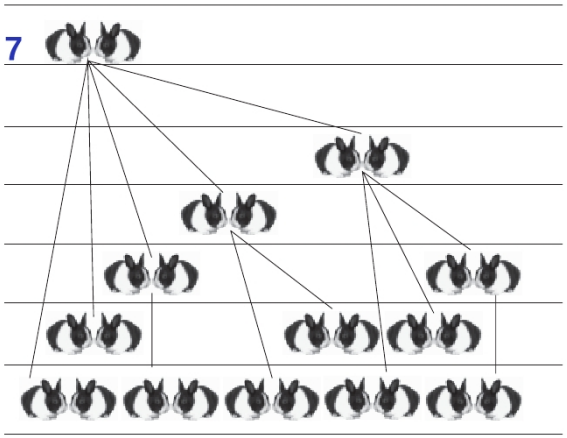
Sea N_t el No. de parejas en cada temporada t: $\Longrightarrow$ $N_{t+1} = N_t + N_{t-1}$, $t = 1, 2, 3, \ldots$.

Si $N_1 = N_2 = 1$ la fórmula anterior genera la famosa sucesión de Fibonacci:

$$1, 1, 2, 3, 5, 8, 13, 21, \ldots \quad ¿\ N_t\ ? \quad \Longrightarrow$$

$$\boxed{N_t = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^t - \left(\frac{1-\sqrt{5}}{2}\right)^t\right] \approx \frac{\left[(2{,}12)^t - (-0{,}12)^t\right]}{2{,}24}}$$

Modelos de poblaciones: Fibonacci 1202



Sea N_t el No. de parejas en cada temporada t: $\Longrightarrow$ $N_{t+1} = N_t + N_{t-1}$, $t = 1, 2, 3, \ldots$.

Si $N_1 = N_2 = 1$ la fórmula anterior genera la famosa sucesión de Fibonacci:

$$1, 1, 2, 3, 5, 8, 13, 21, \ldots \quad ¿\ N_t\ ? \quad \Longrightarrow$$

$$\boxed{N_t = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^t - \left(\frac{1-\sqrt{5}}{2}\right)^t\right] \approx \frac{\left[(2{,}12)^t - (-0{,}12)^t\right]}{2{,}24}}$$

Si $t \gg 1 \Longrightarrow N_t \approx \dfrac{1}{\sqrt{5}}\left(\dfrac{1+\sqrt{5}}{2}\right)^{t+1}$ $\qquad \dfrac{N_{t+1}}{N_t} \approx \dfrac{1+\sqrt{5}}{2}$ **razón áurea**

Veamos como definir con MAXIMA las sucesiones por recurrencia (se recomienda ejecutar el comando `kill(all)` para limpiar la memoria):

```
(%i1) fib[1]:1;fib[2]:1;
fib[n]:=fib[n-1]+fib[n-2];
(%o3) fib[n]:=fib[n−1]+fib[n−2]
```

Veamos como definir con MAXIMA las sucesiones por recurrencia (se recomienda ejecutar el comando `kill(all)` para limpiar la memoria):

```
(%i1) fib[1]:1;fib[2]:1;
fib[n]:=fib[n-1]+fib[n-2];
(%o3) fib[n]:=fib[n−1]+fib[n−2]
```

Luego definimos las listas de n y $fib(n)$

```
(%i4) listat:makelist(n,n,1,15);
      lista1:makelist(fib[n],n,1,15);
      lista:makelist([n,fib[n]],n,1,15);
(%o4) [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]
(%o5) [1,1,2,3,5,8,13,21,34,55,89,144,233,377,610]
```

y las representamos gráficamente:

```
(%i6) wxplot2d([discrete,listat,lista1] ,[style, points]);
```

Claramente es creciente y no acotada (¿por qué?)

Con MAXIMA podemos resolver el sistema analíticamente:

```
(%i7) /* Paquete solve_rec, pag 807 del manual en ES */
      load("solve_rec")$
(%i8) ec:fib1[n]=fib1[n-1]+fib1[n-2]$
(%i9) solve_rec(ec,fib1[n]);
(%o9) fib1[n]=((sqrt(5)-1)^n*%k[1]*(-1)^n)/2^n+
              ((sqrt(5)+1)^n*%k[2])/2^n
(%i10) define(fiba[n],second(solve_rec(ec,fib1[n],
                                fib1[1]=1,fib1[2]=1)))$
(%i11) fiba[n];
(%o11) (sqrt(5)+1)^n/(sqrt(5)*2^n)-
              ((sqrt(5)-1)^n*(-1)^n)/(sqrt(5)*2^n)
```

Con MAXIMA podemos resolver el sistema analíticamente:

```
(%i7) /* Paquete solve_rec, pag 807 del manual en ES */
      load("solve_rec")$
(%i8) ec:fib1[n]=fib1[n-1]+fib1[n-2]$
(%i9) solve_rec(ec,fib1[n]);
(%o9) fib1[n]=((sqrt(5)-1)^n*%k[1]*(-1)^n)/2^n+
              ((sqrt(5)+1)^n*%k[2])/2^n
(%i10) define(fiba[n],second(solve_rec(ec,fib1[n],
                                fib1[1]=1,fib1[2]=1)))$
(%i11) fiba[n];
(%o11) (sqrt(5)+1)^n/(sqrt(5)*2^n)-
              ((sqrt(5)-1)^n*(-1)^n)/(sqrt(5)*2^n)
```

Comparamos la solución exacta y la numérica:

```
(%i12) fiba[3]; radcan(%);
(%o12)(sqrt(5)+1)^3/(8*sqrt(5))+(sqrt(5)-1)^3/(8*sqrt(5))
(%o13) 2
```

Comparamos los 15 primeros términos:

```
(%i14) makelist(radcan(fiba[n]),n,1,15)-lista1;
(%o14) [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
```

Comparamos los 15 primeros términos:

```
(%i14) makelist(radcan(fiba[n]),n,1,15)-lista1;
(%o14) [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
```

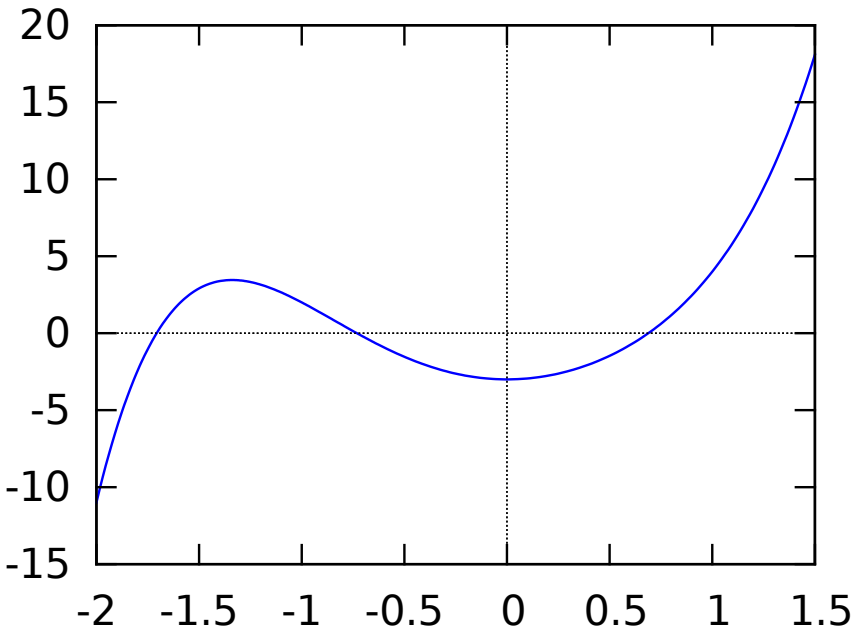
Intentemos calcular el coeficiente $N(n+1)/N(n)$

```
(%i15) define(cociente[n],radcan(fiba[n+1]/fiba[n]));
(%i16) float(cociente[100]);
(%o16) 1.618033988749895
(%i17) float((1+sqrt(5))/2);
(%o17) 1.618033988749895
(%i18) makelist([n,float(cociente[n])],n,1,15)$
(%i19) wxplot2d([discrete,%], [style, points],
                                [point_type, bullet]);
```

¿Es posible caclular este límite con MAXIMA?

Intentar calcular $\lim_{n\to\infty} \dfrac{a^{n+1}+b^{n+1}}{a^n+b_n}$ con $a > b$.

El método de Newton para calcular raíces



Queremos resolver el problema $f(x) = 0$. ¿Cómo proceder?

Queremos resolver el problema $f(x) = 0$. ¿Cómo proceder?

Usamos el Teorema de Bolzano: Método de bisección.

Queremos resolver el problema $f(x) = 0$. ¿Cómo proceder?

Usamos el Teorema de Bolzano: Método de bisección. **Muy lento**

Queremos resolver el problema $f(x) = 0$. ¿Cómo proceder?

Usamos el Teorema de Bolzano: Método de bisección. **Muy lento**

Supongamos que f es *buena* y podemos escribir su polinomios de Taylor en todo un intervalo $[a, b]$ donde sabemos que ha de haber una raíz de f y sea x_0 cierto valor de ese intervalo (si está cerca de la raíz mejor). Entonces

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2}f''(c)(x - x_0)^2, \qquad c \in (x_0, x)$$

Queremos resolver el problema $f(x) = 0$. ¿Cómo proceder?

Usamos el Teorema de Bolzano: Método de bisección. **Muy lento**

Supongamos que f es *buena* y podemos escribir su polinomios de Taylor en todo un intervalo $[a, b]$ donde sabemos que ha de haber una raíz de f y sea x_0 cierto valor de ese intervalo (si está cerca de la raíz mejor). Entonces

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2}f''(c)(x - x_0)^2, \qquad c \in (x_0, x)$$

Supongamos que $z \in (a, b)$ es una raíz de f, entonces

$$0 = f(z) = f(x_0) + f'(x_0)(z - x_0) + \frac{1}{2}f''(c)(z - x_0)^2$$

Pero si x_0 está cerca de z entonces $(z - x_0)^2 \ll |z - x_0|$ y podemos despreciar el último término.

Así

$$0 \approx f(x_0) + f'(x_0)(z - x_0) \quad \Longrightarrow \quad z \approx x_0 - \frac{f(x_0)}{f'(x_0)}$$

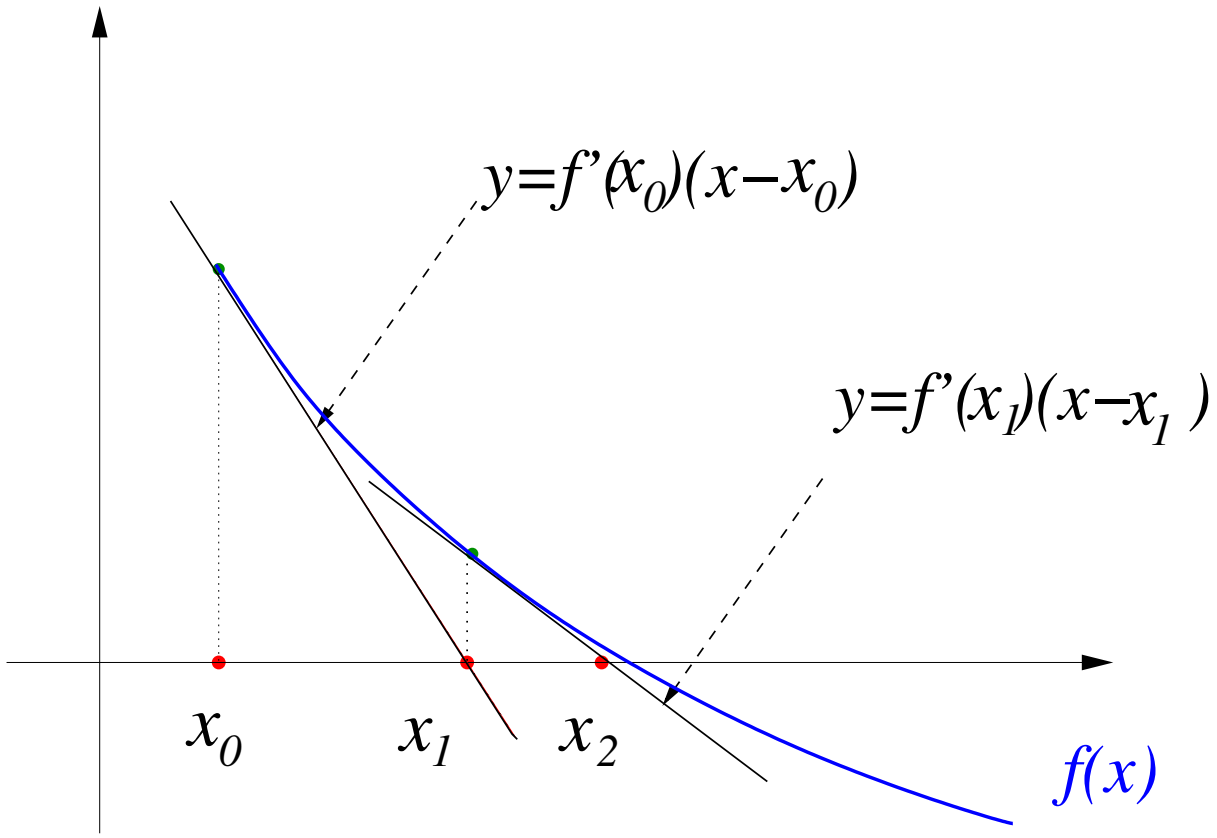
Así

$$0 \approx f(x_0) + f'(x_0)(z - x_0) \quad \Longrightarrow \quad z \approx x_0 - \frac{f(x_0)}{f'(x_0)}$$

Algoritmo:

1. Tomemos un $x_0 \in (a, b)$ tal que $f'(x_0) \neq 0$
2. Definimos la sucesión $x_n = x_{n-1} - \dfrac{f(x_{n-1})}{f'(x_{n-1})}$
3. Calculamos $f(x_n)$ hasta que $|f(x_n)| < \epsilon$, siendo $\epsilon > 0$ el error (tolerancia) del método.

1. Tomemos un $x_0 \in (a, b)$ tal que $f'(x_0) \neq 0$
2. Definimos la sucesión $x_n = x_{n-1} - \dfrac{f(x_{n-1})}{f'(x_{n-1})}$



```
(%i2) kill(p,dp,a)$
      p(x):=x*(x-1)*(x-2);
(%o2) p(x):=x*(x-1)*(x-2)
(%i3) wxplot2d(p(x),[x,-1/2,5/2]);
```

```
(%i2) kill(p,dp,a)$
      p(x):=x*(x-1)*(x-2);
(%o2) p(x):=x*(x-1)*(x-2)
(%i3) wxplot2d(p(x),[x,-1/2,5/2]);

(%o3) (%i4) define(dp(x),diff(p(x),x));
(%o4) dp(x):=(x-1)*x+(x-2)*x+(x-2)*(x-1)
(%i6) a[1]:4$
      a[n]:=a[n-1]-p(a[n-1])/dp(a[n-1]);
```

```
(%i2) kill(p,dp,a)$
      p(x):=x*(x-1)*(x-2);
(%o2) p(x):=x*(x-1)*(x-2)
(%i3) wxplot2d(p(x),[x,-1/2,5/2]);

(%o3) (%i4) define(dp(x),diff(p(x),x));
(%o4) dp(x):=(x-1)*x+(x-2)*x+(x-2)*(x-1)
(%i6) a[1]:4$
      a[n]:=a[n-1]-p(a[n-1])/dp(a[n-1]);

(%i7) a[2];
(%i9) N:10$  lista:makelist(float(a[n]),n,1,N);
(%i10) wxplot2d([[discrete, lista], 2],[x,1,N],[y,1.5,4],
                [style, points, lines])$
(%i14) nn:makelist(n,n,1,N)$
       aa:makelist(float(a[n]),n,1,N)$
       er:makelist(float(2-a[n]),n,1,N)$
       transpose(matrix(nn,aa,er));
```

Ejercicio: Encontrar los ceros de las funciones

1. $f(x) = x^5 + 3x^2 - 1$,
2. $f(x) = 5\log(1 + x^2) - x - 5$,
3. $f(x) = e^x - 3x^2 + 1$.

Procedimiento:

1. Dibujar la función $f(x)$ y establecir distitos intervalos donde haya un cero c.
2. En cada intervalo definir la sucesión por recurrencia con la correspodiente condición inicial.
3. Comprobar si el valor $f(c)$ es suficientemente pequeño según la tolerancia prefijada de antemano.

Algunos modelos de poblaciones

El modelo más sencillo es el modelo de Malthus (1798). Sea una población aislada y

- Sea $p(t)$ el número de individuos en el momento t, $p(t) \gg 1$
- Sea $r(t,p)$ la diferencia entre en índice de natalidad y mortalidad
- La variación $p(t+h) - p(t) \sim p(t)h$ y el coef. de proporcionalidad es $r(t,p)$. Caso más simple $r(t,p) \approx r$.

$$p(t+h) - p(t) = r(t,p)p(t)h \quad \Longrightarrow \quad p_{n+1} = (r+1)p(n).$$

Estudiar los casos cuando $r > 0$ y $r < 0$ ($r > -1$).

Aplicaciones: Modelos sencillo de crecimiento de poblaciones

Algunos modelos de poblaciones

En general los modelos de poblaciones tienen la forma

$$p_{n+1} = rp_n G(p_n, n).$$

Por ejemplo, el modelo de Verhulst (1845)

$$p_{n+1} = \frac{rp_n}{p_n + K}, \quad G(p_n, n) = \frac{1}{p_n + K}, \quad r > K \text{ y } r < K,$$

o el logístico

$$p_{n+1} = rp_n \left(1 - \frac{p_n}{K}\right), \quad G(p_n, n) = 1 - \frac{p_n}{K}.$$

Algunos modelos de poblaciones

En general los modelos de poblaciones tienen la forma

$$p_{n+1} = rp_n G(p_n, n).$$

Por ejemplo, el modelo de Verhulst (1845)

$$p_{n+1} = \frac{rp_n}{p_n + K}, \quad G(p_n, n) = \frac{1}{p_n + K}, \quad r > K \text{ y } r < K,$$

o el logístico

$$p_{n+1} = rp_n \left(1 - \frac{p_n}{K}\right), \quad G(p_n, n) = 1 - \frac{p_n}{K}.$$

Vamos a estudiar numéricamente la solución en ambos casos.

Algunos modelos de poblaciones

- Para el modelo de Verhulst sólo hay dos opciones:

1. si $r > K$ $\lim_{n\to\infty} p(n) = r - K$

```
(%i1) kill(p)$
(%i2) p[1]:2; r:20.0; K:2;
(%i5) p[n]:=r*p[n-1]/(K+p[n-1]);
(%i6) pobm2:makelist(p[k],k,1,10);
(%i7) wxdraw2d(point_type = 7,point_size = 1,
                points(pobm2),yrange = [2,21])$
```

2. Si $r < K$ $\lim_{n\to\infty} p(n) = 0$. Comprobarlo con $r = 20{,}0$ y $K = 200$.

Algunos modelos de poblaciones

- Para el modelo de Verhulst sólo hay dos opciones:

1. si $r > K$ $\lim_{n\to\infty} p(n) = r - K$

```
(%i1) kill(p)$
(%i2) p[1]:2; r:20.0; K:2;
(%i5) p[n]:=r*p[n-1]/(K+p[n-1]);
(%i6) pobm2:makelist(p[k],k,1,10);
(%i7) wxdraw2d(point_type = 7,point_size = 1,
                points(pobm2),yrange = [2,21])$
```

2. Si $r < K$ $\lim_{n\to\infty} p(n) = 0$. Comprobarlo con $r = 20{,}0$ y $K = 200$.

¿Qué está ocurriendo aquí?

Algunos modelos de poblaciones

¿Por qué para el modelo de Verhulst sólo hay dos opciones: o $\lim_{n\to\infty} p(n) = r - K$ si $r > K$ o $\lim_{n\to\infty} p(n) = 0$ si $r < K$?

Algunos modelos de poblaciones

¿Por qué para el modelo de Verhulst sólo hay dos opciones: o $\lim_{n\to\infty} p(n) = r - K$ si $r > K$ o $\lim_{n\to\infty} p(n) = 0$ si $r < K$?

Conviene aquí hacer un estudio teórico para entender que está ocurriendo.

Sucesiones por recurrencia: método general

- Consideremos una sucesión recurrente

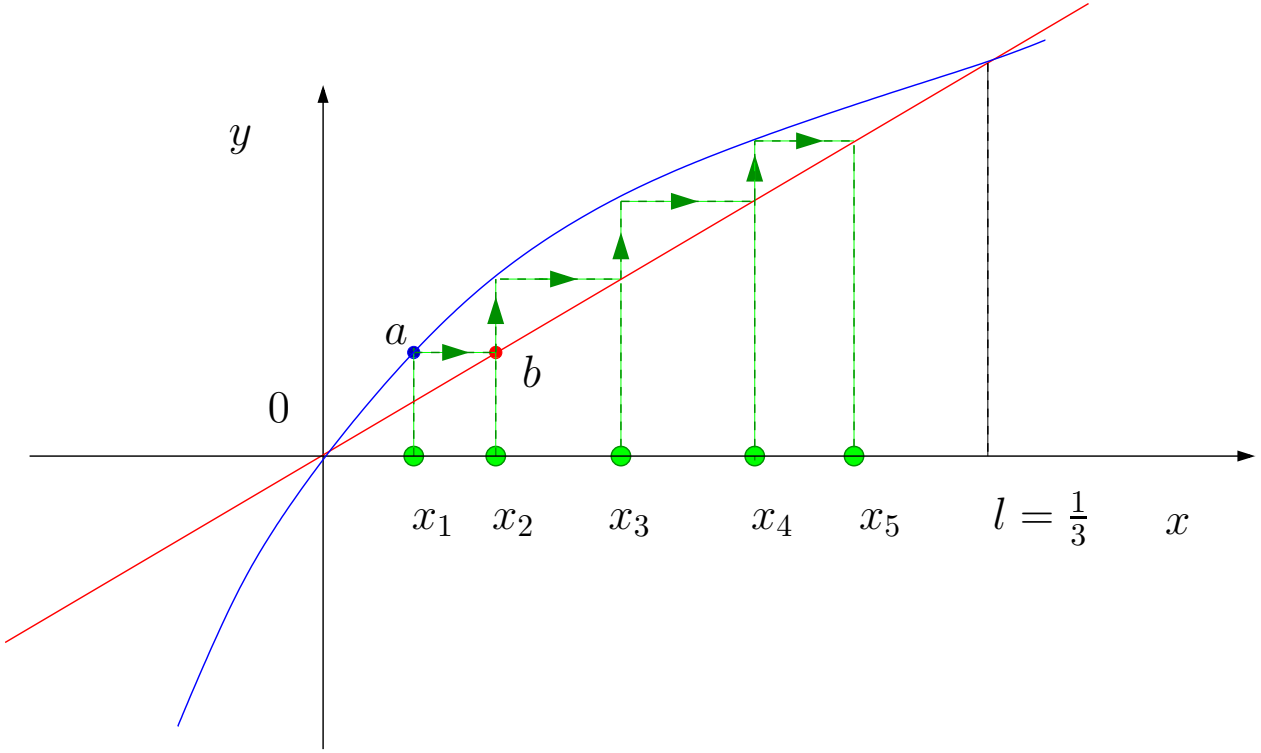
$$x_1 = a, \quad x_{n+1} = f(x_n),$$

donde f es cierta función lo *suficientemente buena*. Nótese que x_n tiene límite, entonces el límite ha de ser la solución de la ecuación $x = f(x)$. A las $x \in \mathbb{R}$ que cumplen lo anterior se les denomina *puntos fijos* de f.

Veamos un método gráfico muy sencillo que nos muestra la tendencia de la sucesión. Lo primero que hacemos es dibujar el gráfico de la función $f(x)$ y la recta $y = x$ (la bisectriz del primer cuadrante).

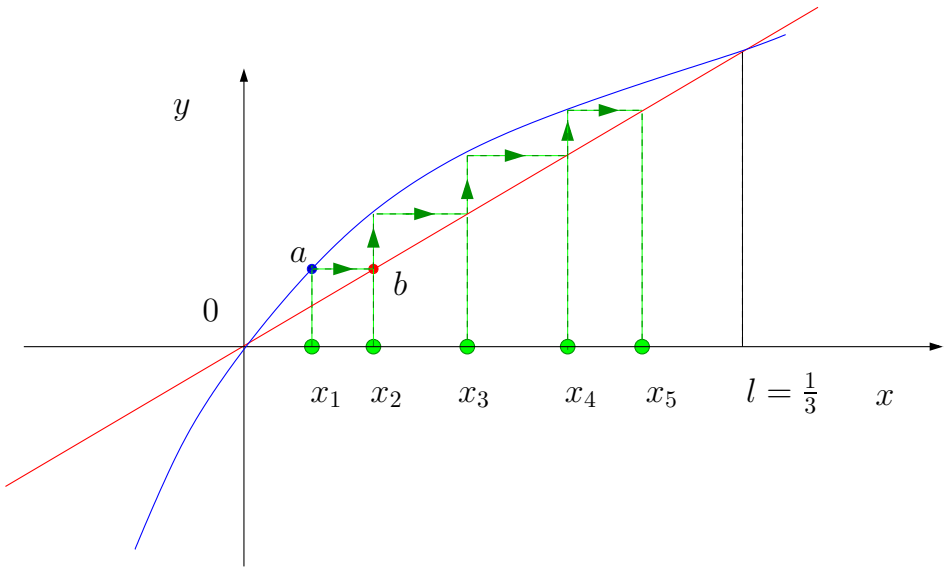
Para aclarar ideas consideremos el ejemplo de la sucesión

$$x_{n+1} = x_n \left(\frac{4}{3} - x_n \right), \quad n \geq 1, \quad x_1 \in \mathbb{R}.$$



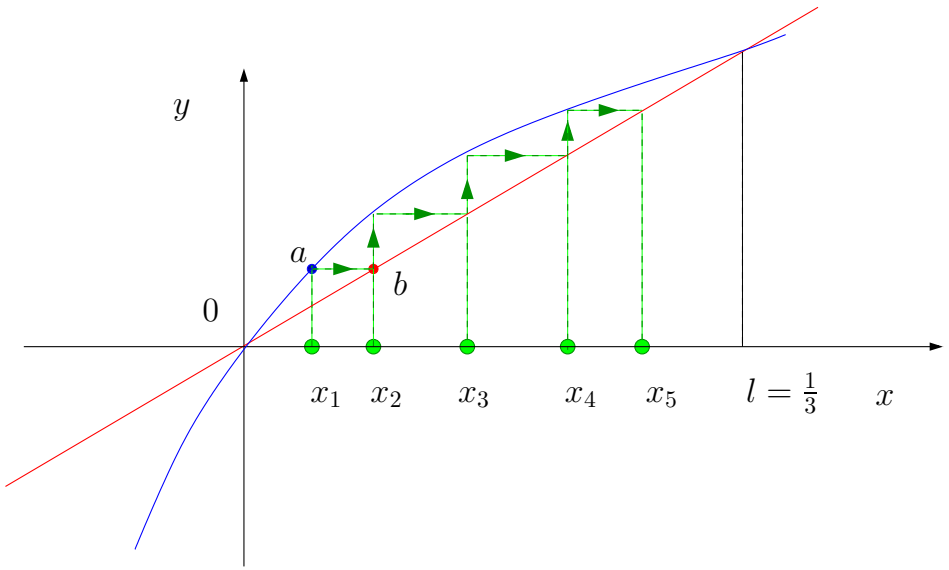
Análisis de la recurrencia $x_{n+1} = x_n \left(\frac{4}{3} - x_n \right)$, $x_1 \in (0, 1/3)$.

```
(%i1) wxplot2d([x*(4/3-x),x],[x,-1,2]);
(%i2) wxplot2d([x*(4/3-x),x],[x,0,0.4],[legend,false]);
```



Análisis de la recurrencia $x_{n+1} = x_n \left(\frac{4}{3} - x_n\right)$, $x_1 \in (0, 1/3)$.

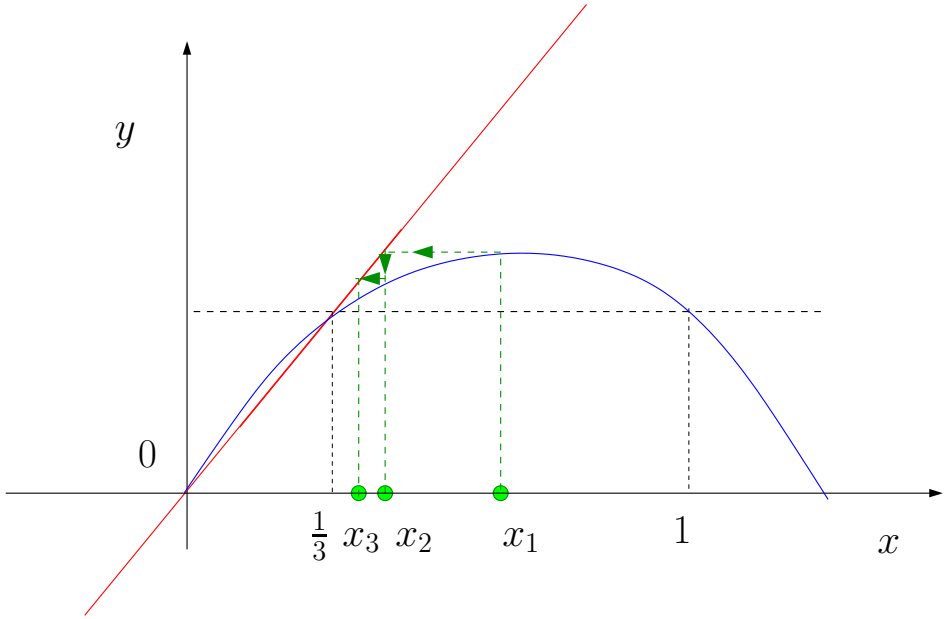
```
(%i3) kill(x)$ x[1]:0.1;
x[n]:=x[n-1]*(4/3-x[n-1]);
makelist(x[k],k,1,4);
lis:makelist(x[k],k,1,30)$
wxplot2d([discrete,lis]);
```



Análisis de la recurrencia $x_{n+1} = x_n \left(\frac{4}{3} - x_n\right)$, $x_1 \in (0, 1/3)$.

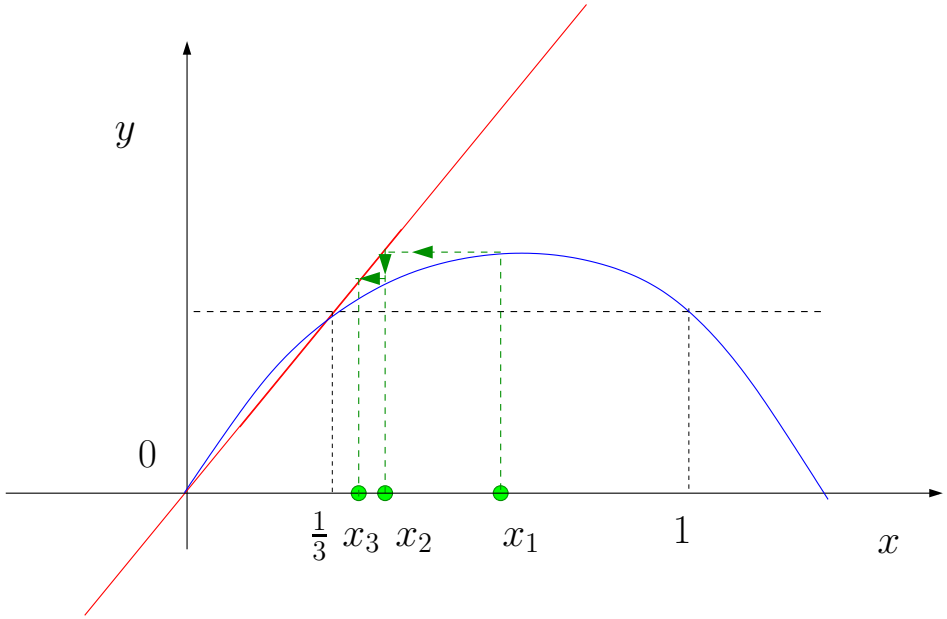
```
(%i3) kill(x)$ x[1]:0.1;
x[n]:=x[n-1]*(4/3-x[n-1]);
makelist(x[k],k,1,4);
lis:makelist(x[k],k,1,30)$
wxplot2d([discrete,lis]);
```

Probemos que x_n es creciente y acotada superiormente.



Análisis de la recurrencia $x_{n+1} = x_n \left(\frac{4}{3} - x_n \right)$, $x_1 \in (1/3, 1)$.

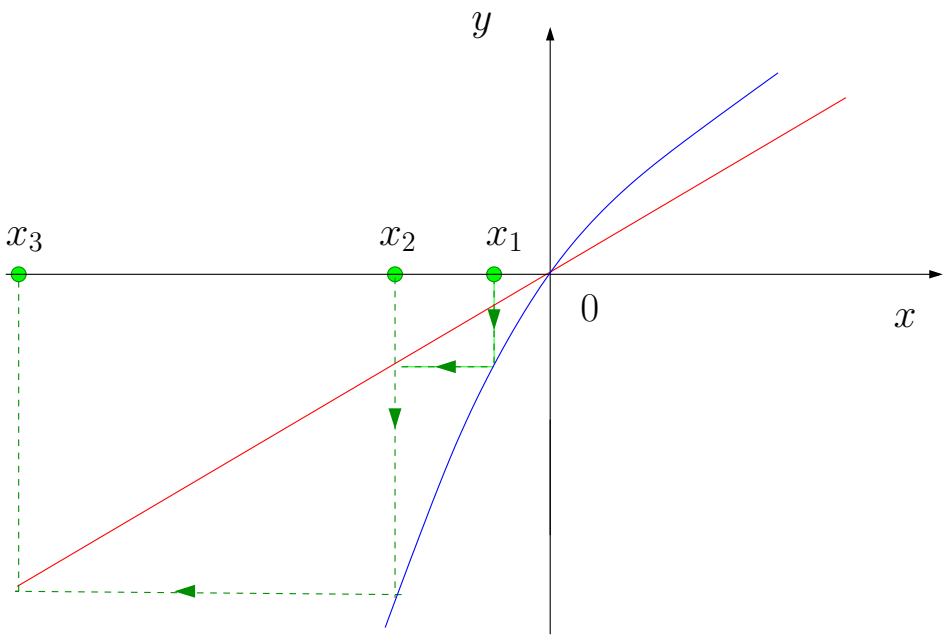
```
(%i9) kill(x)$  x[1]:2/3+.1; /* probar 2/3-0.1 */
x[n]:=x[n-1]*(4/3-x[n-1]);
makelist(x[k],k,1,4);
lis:makelist(x[k],k,1,30)$
wxplot2d([discrete,lis])
```



Análisis de la recurrencia $x_{n+1} = x_n \left(\frac{4}{3} - x_n\right)$, $x_1 \in (1/3, 1)$.

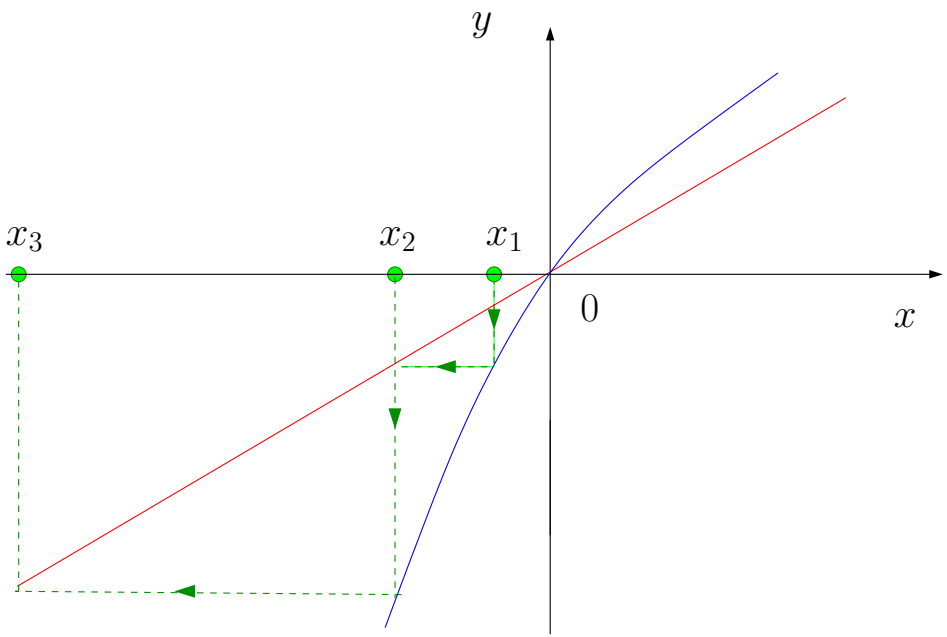
```
(%i9) kill(x)$  x[1]:2/3+.1; /* probar 2/3-0.1 */
x[n]:=x[n-1]*(4/3-x[n-1]);
makelist(x[k],k,1,4);
lis:makelist(x[k],k,1,30)$
wxplot2d([discrete,lis])
```

Probemos que x_n es decreciente y acotada inferiormente.



Análisis de la recurrencia $x_{n+1} = x_n \left(\frac{4}{3} - x_n\right)$, $x_1 < 0$.

```
(%i15)  kill(x)$
x[1]:-0.1; /* probar -1.0 */
x[n]:=x[n-1]*(4/3-x[n-1]);
makelist(x[k],k,1,10);
wxplot2d([discrete,%],[y,-10000,1]);
```



Análisis de la recurrencia $x_{n+1} = x_n \left(\frac{4}{3} - x_n\right)$, $x_1 < 0$.

```
(%i15)  kill(x)$
x[1]:-0.1; /* probar -1.0 */
x[n]:=x[n-1]*(4/3-x[n-1]);
makelist(x[k],k,1,10);
wxplot2d([discrete,%],[y,-10000,1]);
```

Probemos que x_n es decreciente y no acotada inferiormente.

Estudia las recurrencias:

1. $x_1 > 0$, $x_{n+1} = \dfrac{r\,x_n}{x_n + K}$, $r, K > 0$. Elige por ejemplo $K = 10$ y distintos r.

2. $x_1 > 0$, $x_{n+1} = \dfrac{r\,x_n^2}{x_n^2 + K}$, $r, K > 0$. Elige por ejemplo $K = 10$ y distintos r.

Algunos modelos de poblaciones

- El caso del modelo logístico es más complicado. Por sencillez asumiremos $K = 1$.

```
(%i1) r:2; ld[1]:0.4;  ld[n]:=r*ld[n-1]*(1 - ld[n - 1]);
(%o3) ld[n]:=r*ld[n−1]*(1−ld[n−1])
(%i4) pl:makelist(ld[n],n,1,300)$
(%i5) wxplot2d([discrete,pl],[style,points],[y,0.35,0.6],
    [point_type,diamond],[color,blue,red],[legend,false]);
```

También lo podemos dibujar usando el comando `draw2d`

```
load(draw)$
draw2d(point_type=5,point_size=0.5,points(pl))$
```

Algunos modelos de poblaciones

- El caso del modelo logístico es más complicado. Por sencillez asumiremos $K = 1$.

```
(%i1) r:2; ld[1]:0.4;  ld[n]:=r*ld[n-1]*(1 - ld[n - 1]);
(%o3) ld[n]:=r*ld[n−1]*(1−ld[n−1])
(%i4) pl:makelist(ld[n],n,1,300)$
(%i5) wxplot2d([discrete,pl],[style,points],[y,0.35,0.6],
    [point_type,diamond],[color,blue,red],[legend,false]);
```

También lo podemos dibujar usando el comando `draw2d`

```
load(draw)$
draw2d(point_type=5,point_size=0.5,points(pl))$
```

Repetir lo anterior para $r = 3{,}01, 3{,}46, 3{,}56, 3{,}566$

Aplicaciones del Cálculo diferencial

Encontrando ceros de funciones.

Ayudado por los gráficos de las funciones y utilizando el Teorema de Bolzano para las funciones continuas estudia los ceros de las siguientes funciones en $\mathbb{R}$:

1. $x - \sin(x) - 1$
2. $e^x + \sin(x)$
3. $x^{13} - \dfrac{12}{x^2 + 13 + \sin(x)}$

```
(%i1) define(f(x),x-sin(x)-1);
(%i2) wxplot2d([f(x)], [x,-3,5])$
(%i3) r:find_root(f(x), x, 1, 2); f(r);
```

Encontrando ceros de funciones.

Ayudado por los gráficos de las funciones y utilizando el Teorema de Bolzano para las funciones continuas estudia los ceros de las siguientes funciones en $\mathbb{R}$:

1. $x - \sin(x) - 1$
2. $e^x + \sin(x)$
3. $x^{13} - \dfrac{12}{x^2 + 13 + \sin(x)}$

```
(%i1) define(f(x),x-sin(x)-1);
(%i2) wxplot2d([f(x)], [x,-3,5])$
(%i3) r:find_root(f(x), x, 1, 2); f(r);
```

Ejercicios

4. $\arctan(x) + e^x - 15$
5. $\log(x+1) - \cos(x)$

El polinomio de Taylor

Encontrar el polinomio de Taylor de función $f(x) = e^{-x^2}\sin(x)$ alrededor del cero de varios órdenes y dibujarlo junto a la función.

```
(%i1) kill(all)$
      define(f(x),exp(-x^2)*sin(x));
(%i2) wxplot2d(f(x),[x,-5,5]);
(%i3) pt(x,n):=ratexpand(taylor(f(x),x,0,n));
(%i6) k:5$ pt(x,k);
      wxplot2d([f(x),pt(x,k)],[x,-2,2]);
```

Repetir para $k = 5, 10, 15, 20, 30$.

El polinomio de Taylor

Encontrar el polinomio de Taylor de función $f(x) = e^{-x^2}\sin(x)$ alrededor del cero de varios órdenes y dibujarlo junto a la función.

```
(%i1) kill(all)$
      define(f(x),exp(-x^2)*sin(x));
(%i2) wxplot2d(f(x),[x,-5,5]);
(%i3) pt(x,n):=ratexpand(taylor(f(x),x,0,n));
(%i6) k:5$ pt(x,k);
      wxplot2d([f(x),pt(x,k)],[x,-2,2]);
```

Repetir para $k = 5, 10, 15, 20, 30$.

¿Cuántos términos necesitas para aproximar e y e^2 mediante números racionales hasta orden 10^{-4} y 10^{-5}?

El polinomio de Taylor

Encontrar el polinomio de Taylor de función $f(x) = e^{-x^2}\sin(x)$ alrededor del cero de varios órdenes y dibujarlo junto a la función.

```
(%i1) kill(all)$
      define(f(x),exp(-x^2)*sin(x));
(%i2) wxplot2d(f(x),[x,-5,5]);
(%i3) pt(x,n):=ratexpand(taylor(f(x),x,0,n));
(%i6) k:5$ pt(x,k);
      wxplot2d([f(x),pt(x,k)],[x,-2,2]);
```

Repetir para $k = 5, 10, 15, 20, 30$.

¿Cuántos términos necesitas para aproximar e y e^2 mediante números racionales hasta orden 10^{-4} y 10^{-5}?

¿Y para aproximar $\sin(1/2)$ hasta orden 10^{-20}?

Estudiando funciones

- Calcula los extremos de la función $f(x) = \sin((x+1)^2(x+2))$ en el intervalo $[-3, 0]$.

Estudiando funciones

- Calcula los extremos de la función $f(x) = \sin((x+1)^2(x+2))$ en el intervalo $[-3, 0]$.
- Derivando funciones con valores absolutos. Punto crítico: el 0

```
(%i12) wxplot2d(sin(abs(x)),[x,-4*%pi,4*%pi]);
(%i13) diff(sin(abs(x)),x);
(%o13) (x*cos(abs(x)))/abs(x)
(%i14) wxplot2d(%,[x,-4*%pi,4*%pi]);
```

Estudiando funciones

- Calcula los extremos de la función $f(x) = \sin((x+1)^2(x+2))$ en el intervalo $[-3, 0]$.
- Derivando funciones con valores absolutos. Punto crítico: el 0

```
(%i12) wxplot2d(sin(abs(x)),[x,-4*%pi,4*%pi]);
(%i13) diff(sin(abs(x)),x);
(%o13) (x*cos(abs(x)))/abs(x)
(%i14) wxplot2d(%,[x,-4*%pi,4*%pi]);
```

- Calcula los extremos de la función $f(x) = x^x$, $x > 0$. ¿Se podría definir para que fuese continua en $x = 0$?

Estudio de una función "singular".

- Estudiar la función $f : [0, +\infty) \mapsto \mathbb{R}$, $f(x) = x^x$.

```
(%i1) define(f(x),x^x);
(%i2) f(0);
expt: undefined: 0^0
#0: f(x=0) -- an error. To debug this try: debugmode(true);
(%i3) limit(x^x,x,0,plus);
(%o3) 1
(%i4) ff(x):= if equal(x,0) then 1 else x^x;
(%i5) ff(1/2);ff(0);
(%o5) 1/sqrt(2)
(%o6) 1
(%i7) wxplot2d(ff(x),[x,0,1])$
(%i8) diff(f(x),x); define(df(x),ratsimp(%));
(%i10)  solve([second(%)=0], [x]);
(%o10) [x=%e^-1,x^x=0]
(%i11) wxplot2d(df(x),[x,0.1,1])$
```

Estudiando funciones

Estudiar la función $f(x) = (x-3)^2 e^{|x|}$.

```
(%i15) wxplot2d((x-3)^2*exp(abs(x)),[x,-1,4]);
(%i16) diff((x-3)^2*exp(abs(x)),x); df1:factor(%);
(%o17) ((x-3)*%e^abs(x)*(2*abs(x)+x^2-3*x))/abs(x)
(%i18) solve([%=0], [x]);
(%o18) [x=3,x=-sqrt(9-8*abs(x))-3/2,
            x=(sqrt(9-8*abs(x))+3)/2]
```

¿Sólo hay una solución?

Estudiando funciones

Estudiar la función $f(x) = (x-3)^2 e^{|x|}$.

```
(%i15) wxplot2d((x-3)^2*exp(abs(x)),[x,-1,4]);
(%i16) diff((x-3)^2*exp(abs(x)),x); df1:factor(%);
(%o17) ((x-3)*%e^abs(x)*(2*abs(x)+x^2-3*x))/abs(x)
(%i18) solve([%=0], [x]);
(%o18) [x=3,x=-sqrt(9-8*abs(x))-3/2,
            x=(sqrt(9-8*abs(x))+3)/2]
```

¿Sólo hay una solución? Claramente no pues hay un máximo y un mínimo relativo a la derecha de $x = 0$.

Estudiando funciones

Lo anterior es fácil de observar dibujando la derivada:

```
(%i19) wxplot2d(df1,[x,-1/2,3.5]);
```

Y ¿qué pasa con los puntos de inflexión?

```
(%i20) diff((x-3)^2*exp(abs(x)),x,2); df2:factor(%);
(%o21) (%e^abs(x)*(x^2*abs(x)-6*x*abs(x)+
                    11*abs(x)+4*x^2-12*x))/abs(x)
(%i22) wxplot2d(df2,[x,-1/2,3]);
```

¿Cómo proceder en estos casos? Usando la definición de valor absoluto: definiendo una función a trozos.

- Estudiar la función $f(x) = \begin{cases} x + \sqrt{1+x^2}, & \text{si } x \leq 0, \\ \dfrac{1}{1+(x\log(x))^2}, & \text{si } x > 0. \end{cases}$

```
(%i1) h(x):=if x<=0 then x+sqrt(1+x^2)
                    else 1/(1+(x*log(x))^2);
(%i2) define(h1(x),if x<=0 then x+sqrt(1+x^2)
                    else 1/(1+(x*log(x))^2));
(%i3) wxplot2d(h1(x),[x,-5,5],[ylabel,"h(x)"]);
(%i4) diff(h(x),x);
(%i5) ev(%,1);
    ev: improper argument: 1
    -- an error. To debug this try: debugmode(true);
(%i6) h1:diff(x+sqrt(1+x^2),x);
(%i7) h2:diff(1/(1+(x*log(x))^2),x);
```

```
(%i8) define(dh(x),if x<=0 then h1 else h2);
(%o8) dh(x):=if x<=0 then x/sqrt(x^2+1)+1
      else -(2*x*log(x)^2+2*x*log(x))/(x^2*log(x)^2+1)^2
(%i9) dh(x);
```

Comprobamos como funciona:

```
(%i10) assume(a<0)$ h(a);
(%o11) sqrt(a^2+1)+a
(%i12) assume(b>0)$ h(b);
(%o13) 1/(b^2*log(b)^2+1)
(%i14) wxplot2d(dh(x),[x,-4,4]);
(%i15) limit(h1,x,0,minus);
(%o15) 1
(%i16) limit(h2,x,0,plus);
(%o16) 0
```

Buscamos los extremos

```
(%i17) solve(h1=0);
(%o17) [x=-sqrt(x^2+1)]
```

que no tiene sentido (¿por qué?)

```
(%i18) solve(h2,x);
(%o18) [x=%e^-1,x=1,x=0]
```

Luego hay dos posibles extremos. Usamos la 2ª derivada:

```
(%i60) der:diff(h2,x)$ ratsimp(ev(der,sol[1]));
(%o61) (2*%e^4)/(%e^4+2*%e^2+1)
(%i62) ratsimp(ev(der,sol[2]));
(%o62) -2
```

Finalmente, miramos las asíntotas horizontales

```
(%i19) limit(h1 ,x,-inf); limit(h2, x, inf);
(%o19) 0
(%o20) 0
```