Computación de Alto Rendimiento con MATLAB

UC3M Scientific Computing Center C3

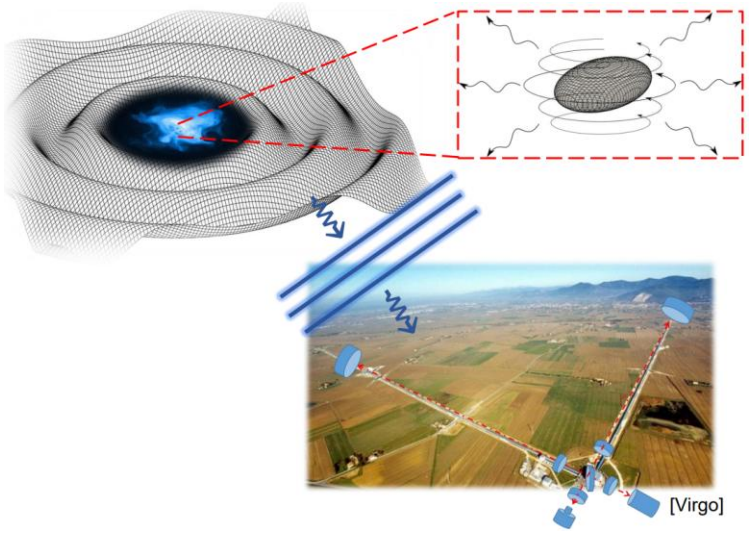
Ángel Sierra
Responsable de
Comunidad Académica

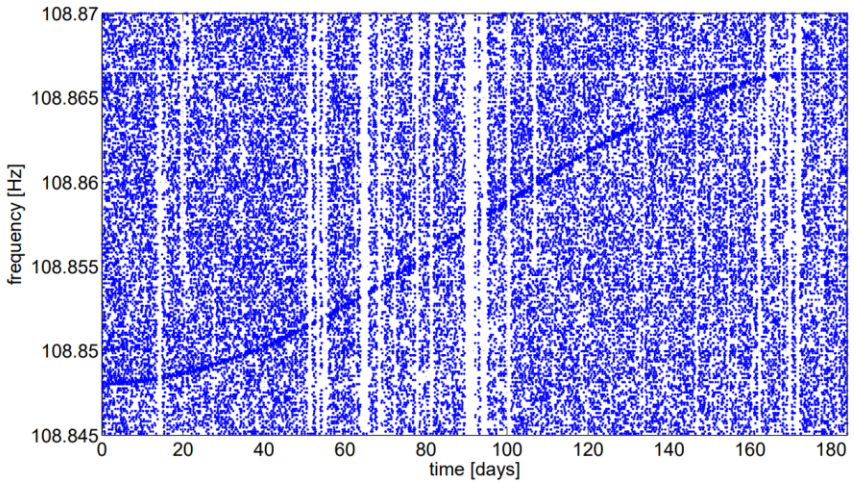
Carlos Sanchis
Customer Success Engineering
Academia, Southern Europe

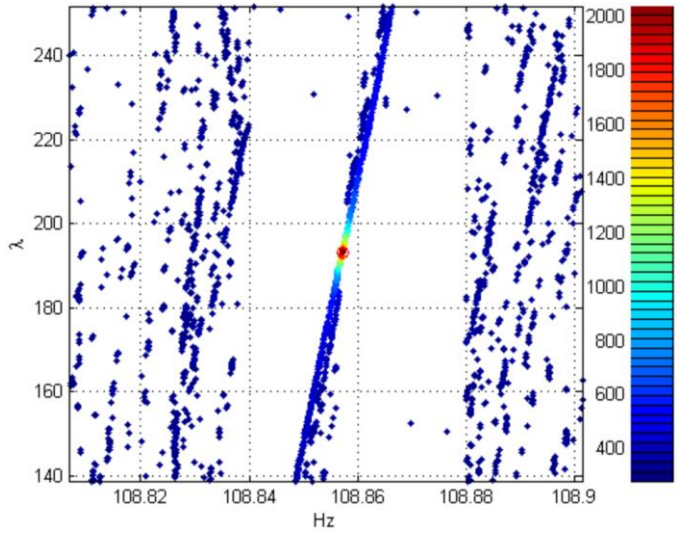
Investigadores del INFN aceleran la búsqueda de ondas gravitacionales usando MATLAB Parallel Server en el supercomputador Leonardo (CINECA)

Pia Astone
INFN

Lorenzo Pierini
INFN







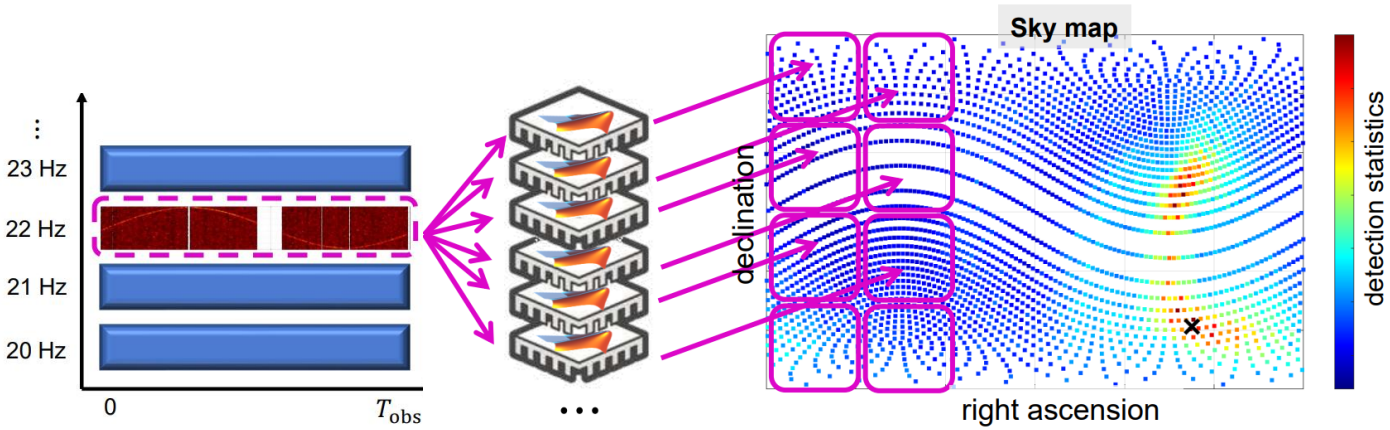
¡10 millones de horas de CPU por detector y año!

Investigadores del INFN aceleran la búsqueda de ondas gravitacionales usando MATLAB Parallel Server en el supercomputador Leonardo (CINECA)

Pia Astone
INFN

Lorenzo Pierini
INFN

Alessio Conte
MathWorks



Paralelización
Reducción a 9 meses

GPUs
Reducción a 4 semanas

Customer Success Engineering
Academia, Southern Europe


Carlos Sanchis


Paolo Panarese


Alessio Conte

INFN


Paula Poza


Alberto Álvarez


Jennifer Gago

uc3m


Charbel Cherfan


Daniele Sportillo

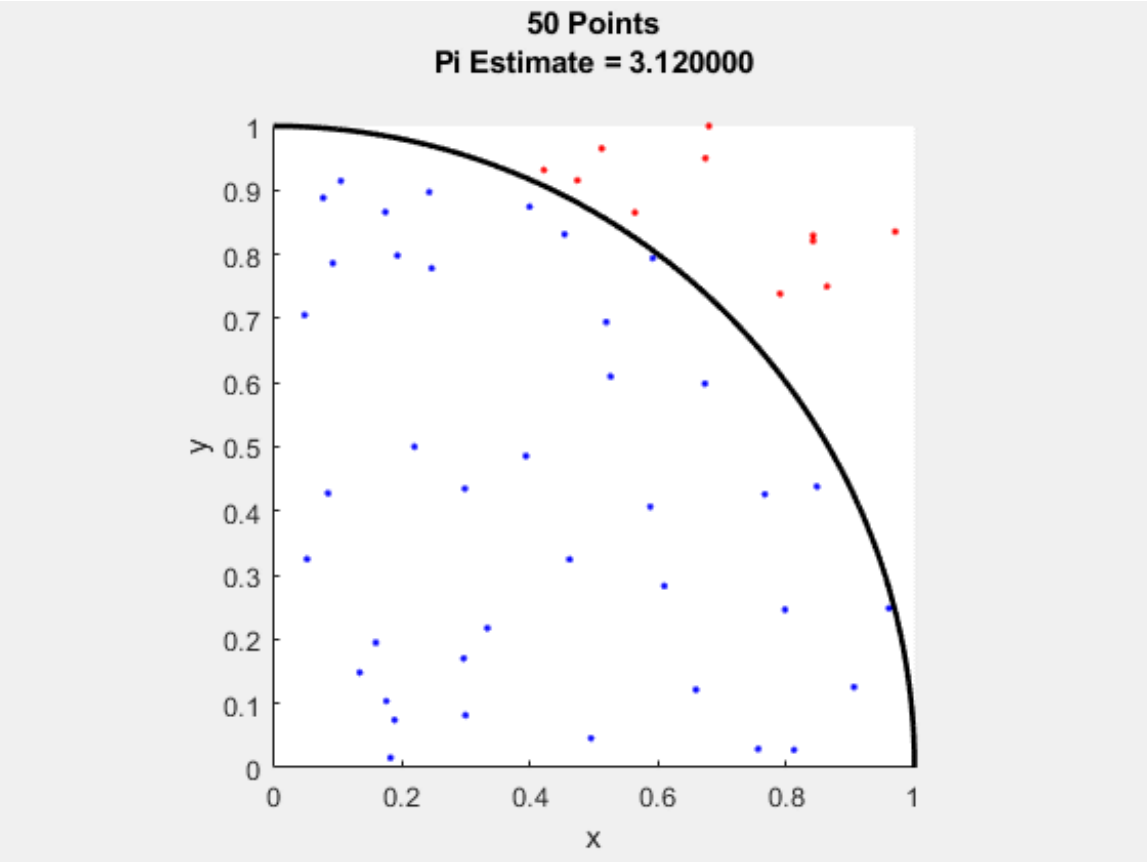

Nadia Bedjaoui

Computación de Alto Rendimiento con MATLAB

1. Acelerar el código en MATLAB
2. Paralelizar con Parallel Computing Toolbox
3. Usar GPUs NVIDIA sin escribir CUDA
4. Lanzar trabajos a un clúster con MATLAB Parallel Server
5. Paralelización más allá de MATLAB

Ejemplo: aproximación de π

1. Generar un gran número (N) de puntos aleatorios $(\mathrm{x}, y) \in [0,1]$
2. Contar cuántos (Z) quedan dentro de la circunferencia $\mathrm{x}^2 + \mathrm{y}^2 \leq 1$
3. Estimación: $\pi \approx 4\frac{Z}{N}$



Hardware…

Código CPU

- Amazon AWS M6a
- 32 physical cores (64 v CPUs)
- 256Gb RAM

Código GPU

- NVIDIA RTX 3070 (local machine)
- NVIDIA A100 on Karolina (Czech National Super Computer)

Ejemplo: aproximación de π

```
count = 0;

for i = 1:N
    count = count +
monteIteration();
end

piEst = 4*count/N;

function iter = monteIteration()
    x = rand();
    y = rand();
    iter = 0;
    z = x*x + y*y;
    if z < 1
        iter = 1;
    end
end
```

```
Estimate for pi is 3.14259080 after 0.692839 seconds
Absolute error is 9.981e-04
14.43 million samples per second
```

¡80 días para encontrar 7 decimales!

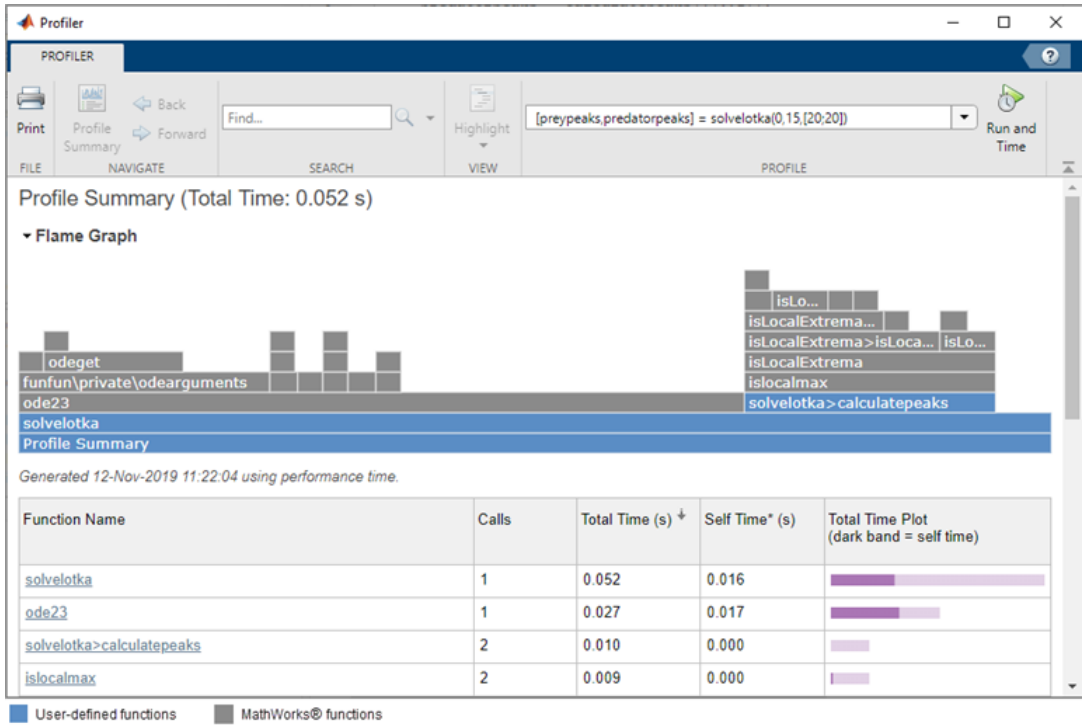
1. Acelerar el código en MATLAB

- Medir el tiempo de ejecución

tic toc
(stopwatch timer)

```
tic
count = 0;

for i = 1:N
    count = count + monteIteration();
end

piEst = 4*count/N;
toc
```

MATLAB Profiler



>> www.mathworks.com/help/matlab/matlab_prog/profiling-for-improving-performance.html

1. Acelerar el código en MATLAB

- Medir el tiempo de ejecución
- Preasignación

```
count = 0;

x = rand(N,1);
y = rand(N,1);

for i = 1:N
    if (x(i)*x(i) + y(i)*y(i)) < 1
        count = count + 1;
    end
end

piEst = 4*count/N;
```

```
Estimate for pi is 3.14224160 after 0.229212 seconds
Absolute error is 6.489e-04
43.63 million samples per second
```

27 días

1. Acelerar el código en MATLAB

- Medir el tiempo de ejecución
- Preasignación
- Vectorización

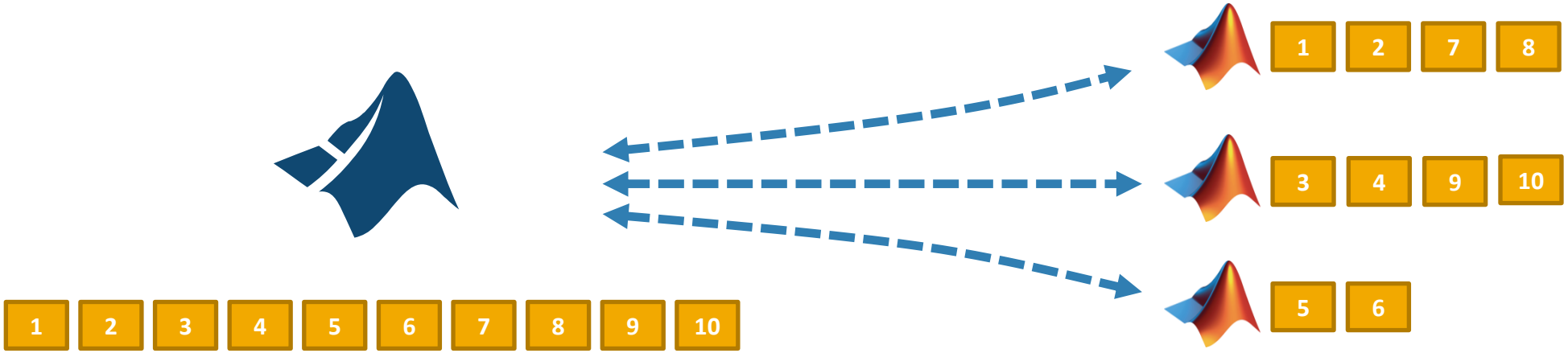
```
x = rand(N,1);
y = rand(N,1);

count = sum((x.*x + y.*y) <= 1);

piEst = 4*count/N;
```

```
Estimate for pi is 3.14179400 after 0.176337 seconds
Absolute error is 0.00020135
56.71 million samples per second
```

20 días

>> www.mathworks.com/help/matlab/matlab_prog/techniques-for-improving-performance.html

2. Paralelizar con Parallel Computing Toolbox




```
a = zeros(10, 1);
b = pi;
for i = 1:10
   a(i) = i + b;
end
disp(a)
```

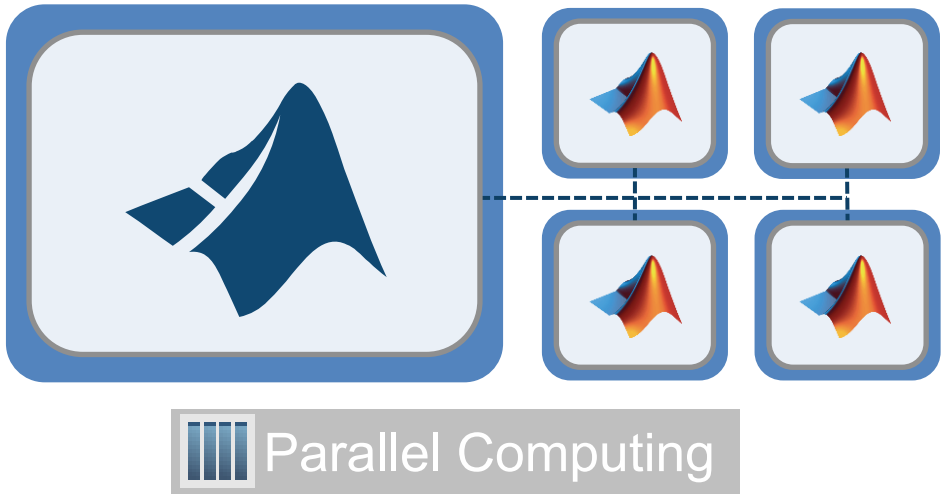
```
a = zeros(10, 1);
b = pi;
parfor i = 1:10
   a(i) = i + b;
end
disp(a)
```

>> www.mathworks.com/help/coder/ug/classification-of-variables-in-parfor-loops.html

2. Paralelizar con Parallel Computing Toolbox

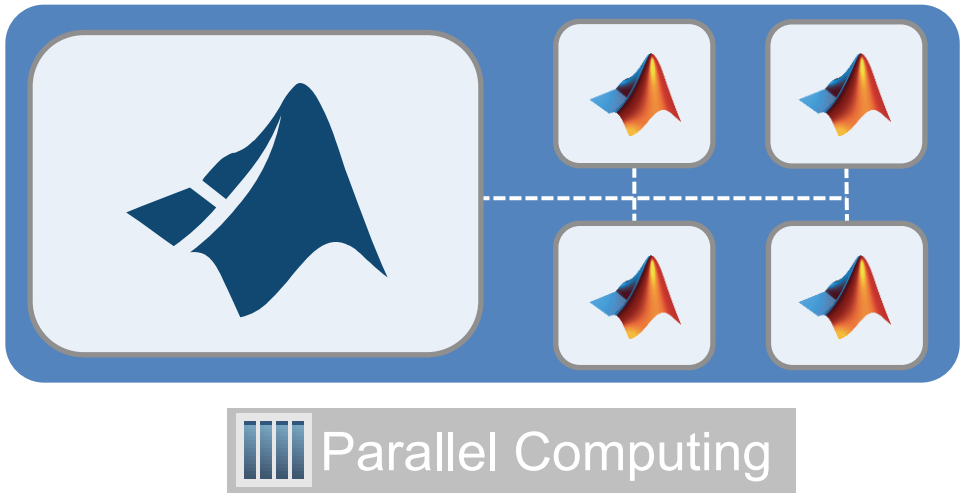
```
>> parpool("Processes")
```

Workers run as their own process




```
>> parpool("Threads")
```

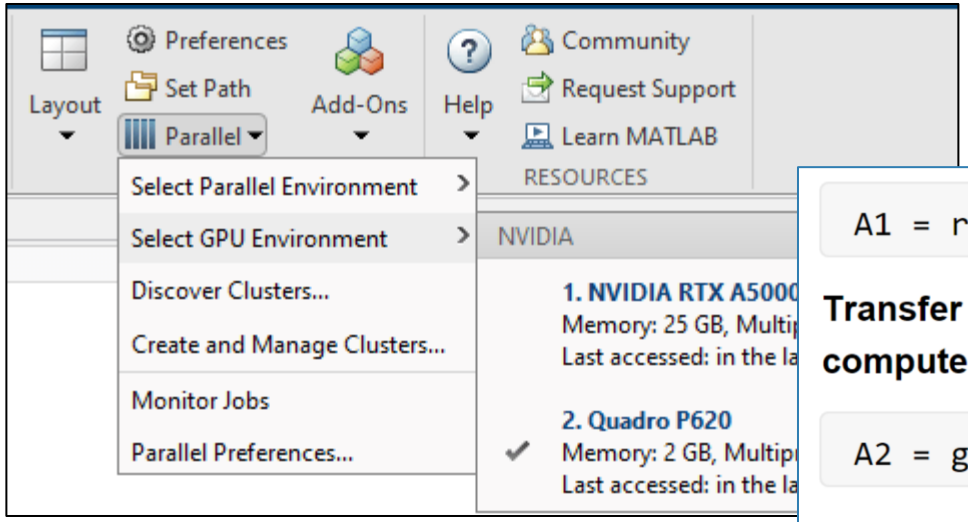
Workers run as threads in main MATLAB process and share memory




```
>> https://www.mathworks.com/help/parallel-computing/
choose-between-thread-based-and-process-based-environments.html
```

3. Usar GPUs NVIDIA sin escribir CUDA

MATLAB with
Parallel Computing Toolbox




```
A1 = rand(3000,3000);
```

Transfer data to GPU from computer memory

```
A2 = gpuArray(A1);
```

Perform calculation on GPU

```
B2 = fft(A2);
```

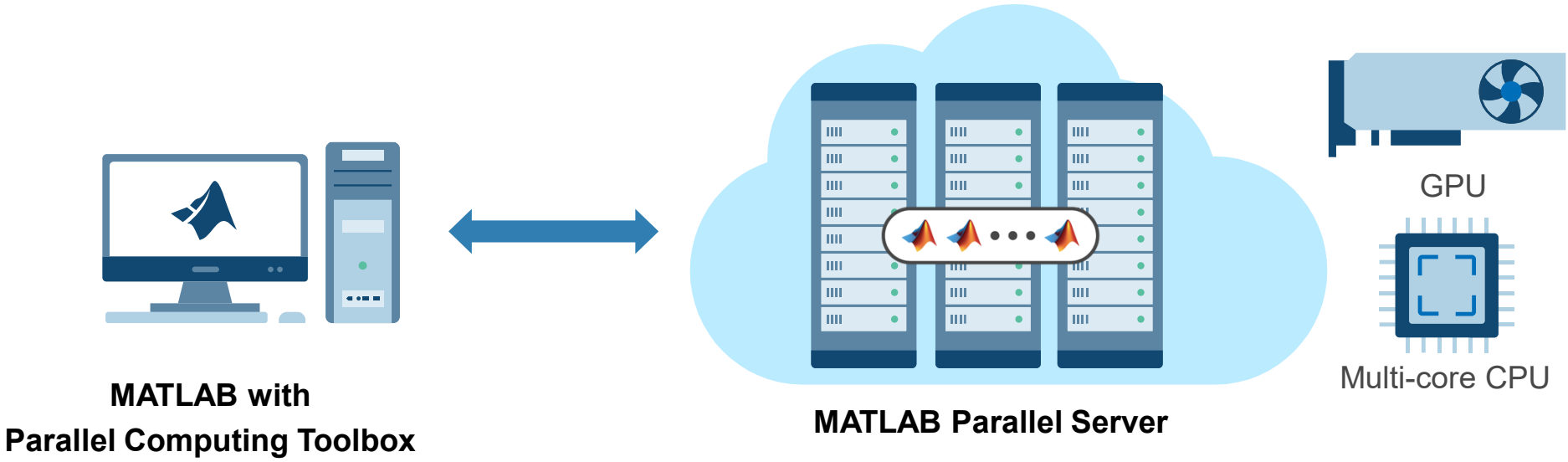
Gather data or plot

```
B2=gather(B2);
```

- *gpuArray* y *gather* para intercambiar datos explícitamente
- Muchas funciones de MATLAB tienen soporte para GPUs
- Soporte para multi-GPU

```
>> https://www.mathworks.com/solutions/gpu-computing.html
```

4. Lanzar trabajos a un clúster con MATLAB Parallel Server



- Desarrollo en ordenador de sobremesa
- Integración con los recursos y datos del cluster

```
>> https://c3-uc3m.github.io/c3-web/user_docs/matlab
```

Ejemplo: aproximación de π

Idiomatic MATLAB

Version	Speed (samples/s)	Time to 7 digits accuracy (10^{14} samples)
Initial version	14.43 million	80.21 days
Inline the Monte-Carlo function	17.23 million	67 days
Vector of random numbers	38.39 million	30 days
Remove un-necessary array	43.63 million	27 days
Fully vectorized	56.71 million	20.4 days

Changing Random Number Generation Algorithms

Version	Speed (samples/s)	Time to 7 digits accuracy (10^{14} samples)
Fast Mersenne Twister	117.2 million	9.88 days
Threefry (Implicitly parallel)	1,208 million	0.96 days

Explicitly parallel using parfor

Version	Speed (samples/s)	Time to 7 digits accuracy (10^{14} samples)
Local parfor using parpool("threads")	3,694 million	7.5 hours
parfor: 4 node cluster of 32 cores each	13,700 million	2 hours

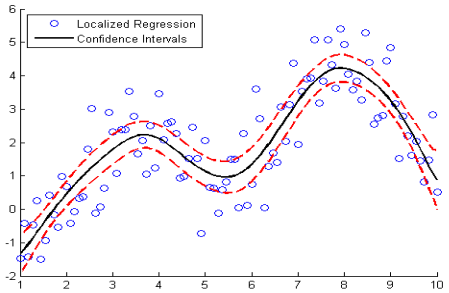
GPU Computing

Version	Speed (samples/s)	Time to 7 digits accuracy (10^{14} samples)
Do everything on the GPU (Desktop GPU: RTX 3070)	10,271 million	2.7 hours
Single precision (Desktop GPU: RTX 3070)	12,828 million	2.1 hours
Arrayfun(Desktop GPU: RTX 3070)	19,569 million	85 minutes
Arrayfun (Karolina GPU: NVIDIA A100)	50,815 million	32 minutes
Multi-GPU Using 8 GPUs on Karolina	384,791.5 million	4.3 minutes

5. Paralelización más allá de MATLAB...

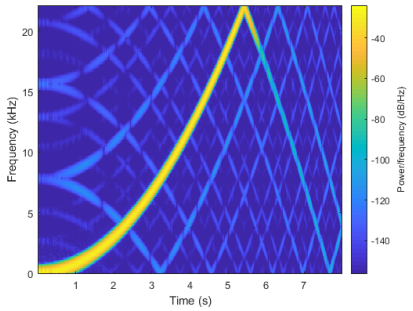
Image Processing

Statistics and Machine Learning

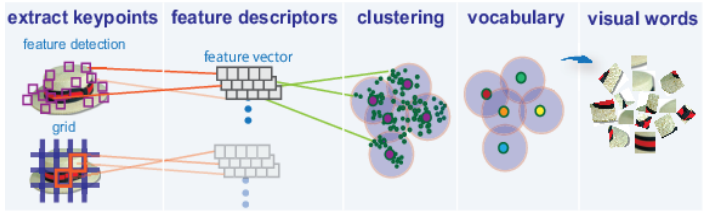


Deep Learning

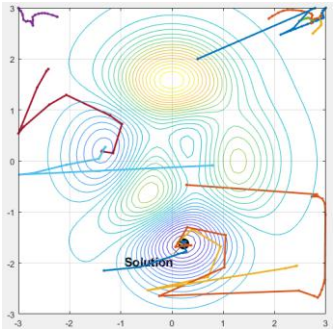
Signal Processing & Communications



Computer Vision



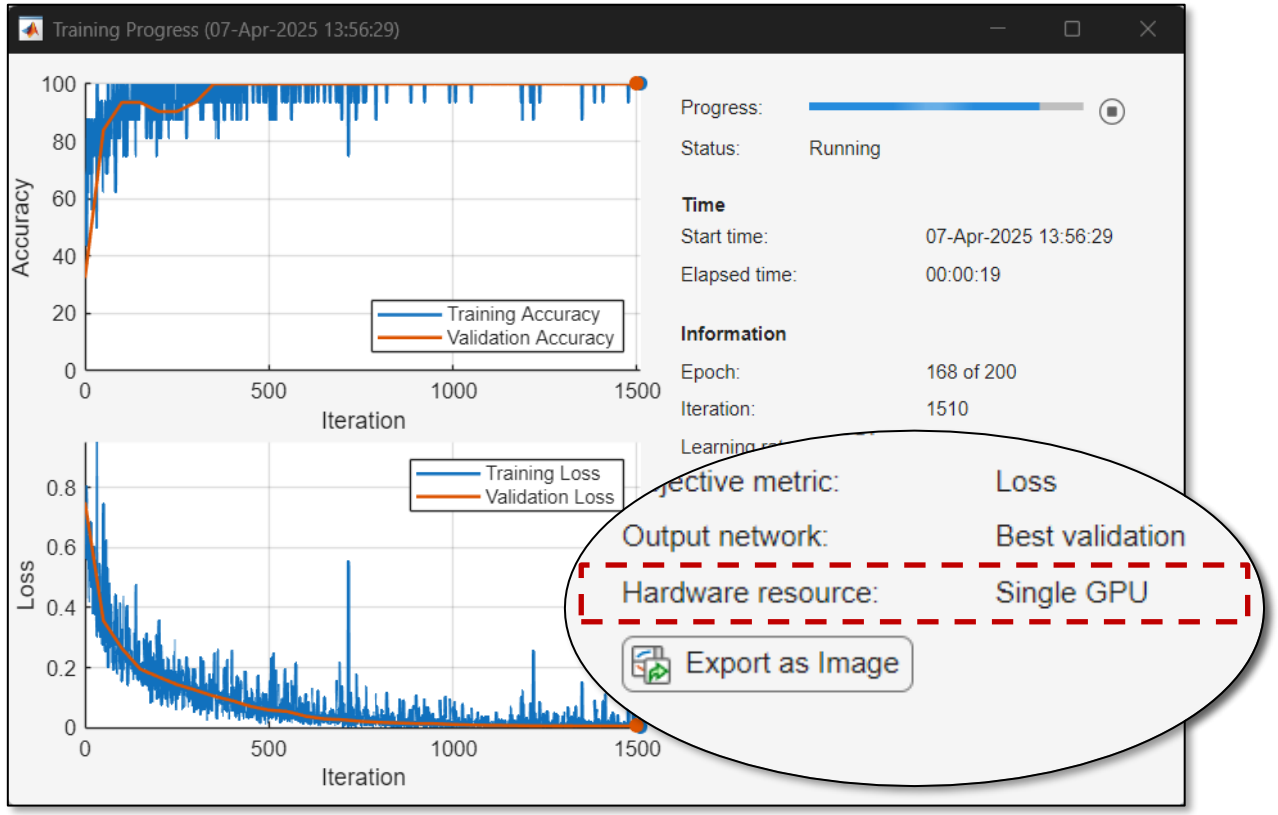
Optimization & Global Optimization



>> https://www.mathworks.com/products/parallel-computing/parallel-support.html

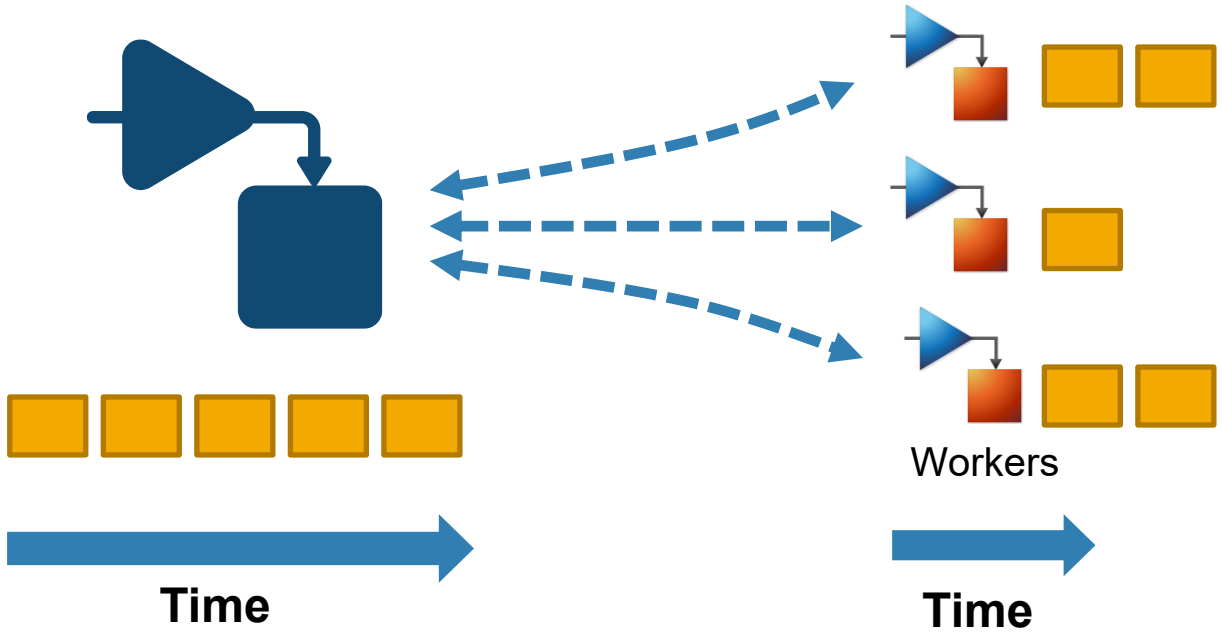
Entrenamiento de redes neuronales con GPUs

```
options = trainingOptions("adam", ...
    MaxEpochs = 200, ...
    ExecutionEnvironment = "gpu", ...
    PreprocessingEnvironment = background, ...
    MiniBatchSize = 16, ...
    Shuffle = "every-epoch", ...
    ValidationData = tblValidation, ...
    Plots = "training-progress", ...
    Metrics = "accuracy", ...
    OutputNetwork = "best-validation", ...
    Verbose = false);
```

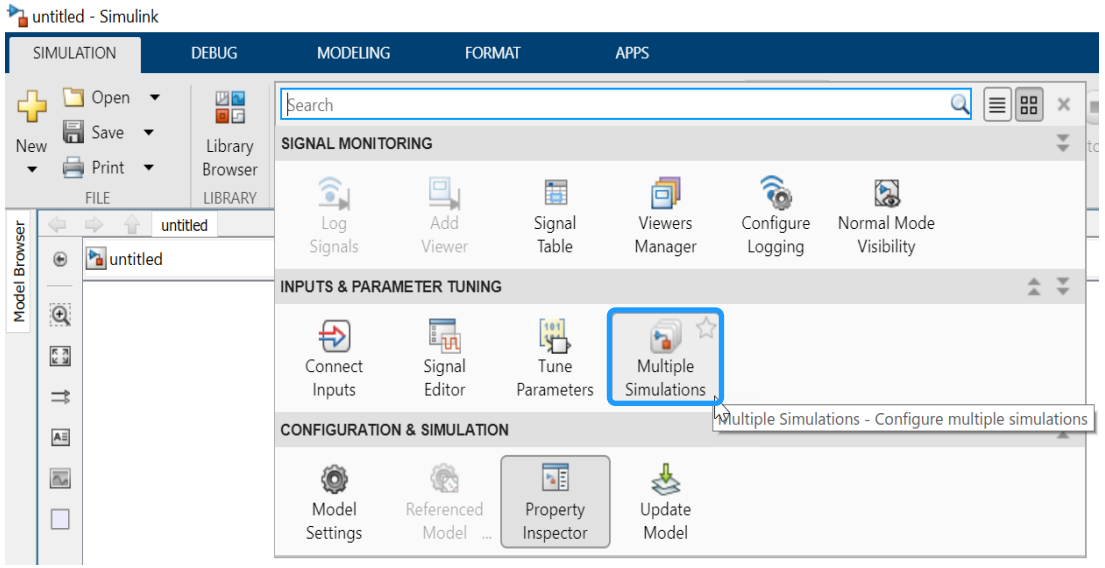


>> www.mathworks.com/help/deeplearning/ug/deep-learning-with-matlab-on-multiple-gpus.html

Simulaciones en paralelo




```
for i = 10000:-1:1
    in(i) = Simulink.SimulationInput(my_model);
    in(i) = in(i).setVariable(my_var, i);
end
out = parsim(in);
```



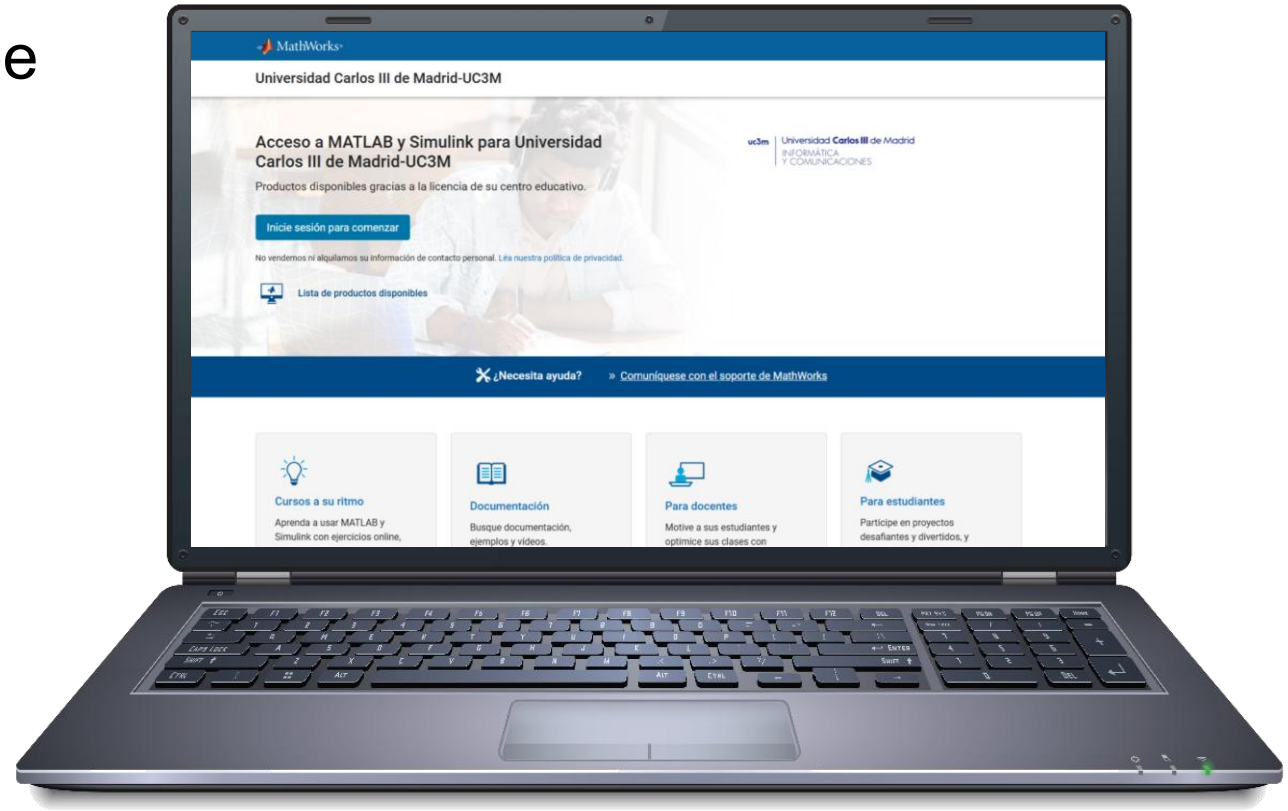
Computación de Alto Rendimiento con MATLAB

1. Acelerar el código en MATLAB
2. Paralelizar con Parallel Computing Toolbox
3. Usar GPUs NVIDIA sin escribir CUDA
4. Lanzar trabajos a un clúster con MATLAB Parallel Server
5. Paralelización más allá de MATLAB

Primeros pasos…

Campus-Wide Platform

- MATLAB, Simulink y más de 100 toolboxes, en cualquier dispositivo
- Cursos online interactivos
- Corrección automática de ejercicios
- Software para servidores y en la nube
- Soporte técnico y académico

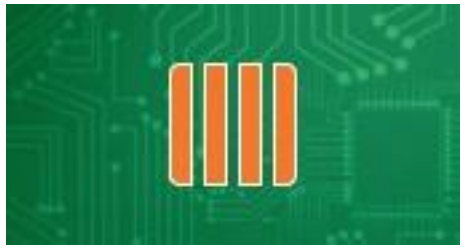


>> https://matlab.mathworks.com

Formación online interactiva

MATLAB Onramp
Get started with the MATLAB language and environment so that you can analyze science and engineering data.

MATLAB Coding Practices for Efficiency and Performance
Write efficient and performant code. Learn how to identify bottlenecks, increase efficiency through vectorization and preallocation, and examine memory usage.



Parallel Computing Onramp
Learn to speed up your code by using multiple CPU cores to run for-loops in parallel and writing code that handles information efficiently.

```
>> https://matlabacademy.mathworks.com
```

Customer Success Engineers

Asesoramos gratuitamente a docentes e investigadores
en proyectos con MATLAB & Simulink.

Jennifer Gago

jgagomu@mathworks.com

¡Ponte en contacto con nosotros!