

# Tema 4. Análisis comparativo del rendimiento

*¿Qué servidor tiene mejor rendimiento?*

Analistas, administradores y diseñadores



# Objetivos del tema

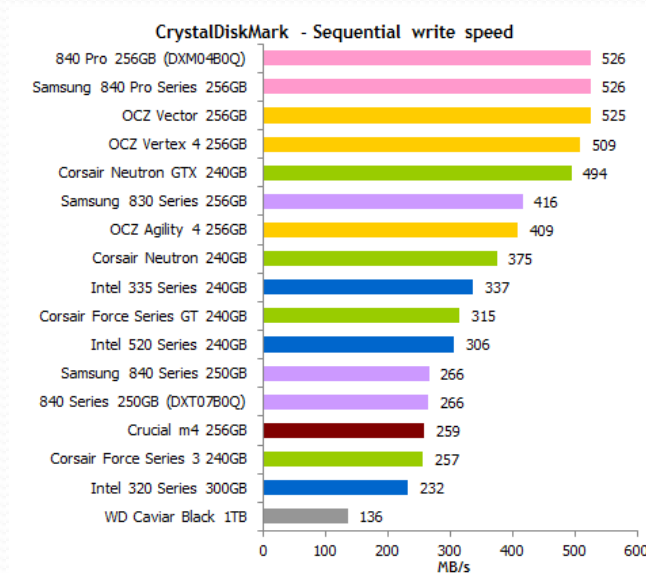
- Entender la problemática inherente al diseño de un índice de rendimiento cualquiera.
- Interpretar los índices clásicos de rendimiento usados en el ámbito de los procesadores.
- Entender el concepto de benchmark y sus distintos tipos.
- Conocer ejemplos reales de benchmarks.
- Conocer diferentes estrategias de análisis para hacer comparaciones de rendimiento así como las condiciones para hacer una comparación de rendimiento lo más ecuánime posible.

# Bibliografía

- *Evaluación y modelado del rendimiento de los sistemas informáticos*. Xavier Molero, C. Juiz, M. Rodeño. Pearson Educación, 2004. Capítulo 3.
- *Measuring computer performance: a practitioner's guide*. David J. Lilja, Cambridge University Press, 2000. Capítulos 2,5 y 7.
- *The art of computer systems performance analysis : Techniques for experimental design, measurement, simulation, and modeling*. Raj Jain, John Wiley & Sons, 1991. Capítulos 9, 13 y 20.
- *System Performance Tuning*. Gian-Paolo D. Musumeci, Mike Loukides, 2nd Edition - O'Reilly Media, 2002. Capítulo 2.
- *The Standard Performance Evaluation Corporation (SPEC)*, <http://www.spec.org>.
- *The Transaction Processing Performance Council (TPC)*, <http://www.tpc.org>.

# Contenido

- Referenciación (benchmarking).
- Análisis de los resultados de un test de rendimiento.
- Comparación de prestaciones en presencia de aleatoriedad.
- Diseño de experimentos de comparación de rendimiento.



## 4.1. Referenciación (*Benchmarking*)

# Características de un buen índice de rendimiento de un sistema informático

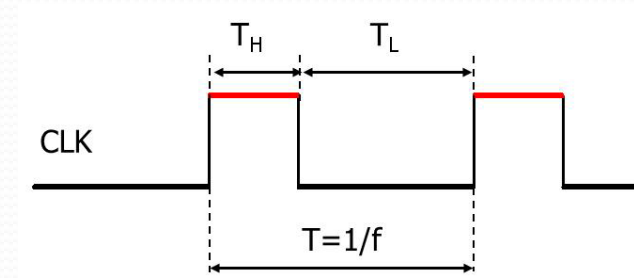
- **Representatividad y fiabilidad:** Si un sistema A siempre presenta un índice de rendimiento mejor que el sistema B, es porque **siempre** el rendimiento real de A es mejor que el de B.
- **Repetibilidad:** Siempre que se mida el índice en las mismas condiciones, el valor de éste debe ser el mismo.
- **Consistencia y facilidad de medición:** El índice se debe poder medir en cualquier sistema informático y esta medida debe ser fácil de tomar.
- **Linealidad:** Si el índice de rendimiento aumenta, el rendimiento real del sistema debe aumentar en la misma proporción.





# Tiempo de ejecución, frecuencia de reloj y CPI

¿Pueden ser la frecuencia de reloj ( $f_{\text{RELOJ}}$ ) o el número medio de ciclos por instrucción (**CPI**) buenos índices de rendimiento?



$$T_{EJEC} = NI \times CPI \times T_{RELOJ} = \frac{NI \times CPI}{f_{RELOJ}}$$

- $T_{EJEC}$  = Tiempo de ejecución del programa.
- NI = Número de instrucciones del programa.

- No lo son. Es posible encontrar ejemplos de sistemas con  $f_{\text{RELOJ}}$  (o CPI) peores que otros pero con mejores prestaciones.
- ¿Y si usamos directamente el tiempo de ejecución ( $T_{EJEC}$ ) de un determinado programa?
  - ¿Consistencia? El programa debería estar descrito en un lenguaje de alto nivel.
  - ¿Repetibilidad? El programa debería ejecutarse en un entorno muy controlado.
  - ¿Representatividad y fiabilidad? Dependería del programa a ejecutar.

# MIPS (million of instructions per second)

- En principio, parece una medida prometedora ya que representa cómo de rápido ejecuta las instrucciones un microprocesador.

$$MIPS = \frac{NI}{T_{EJEC} \times 10^6} = \frac{f_{RELOJ}}{CPI \times 10^6}$$

Inconvenientes:

- Depende del juego de instrucciones (ej. RISC vs CISC).
- Además, los MIPS medidos varían incluso entre programas en el mismo computador.

```
def add_forty_two(n)
  pushq %rbp
  movq %rsp, %rbp
  addl $42, %edi
  movl %edi, %eax
  popq %rbp
  retq
end
```

```
slli x30, x5, 3 // x30 = f*8
add x30, x10, x30 // x30 = &A[f]
slli x31, x6, 3 // x31 = g*8
add x31, x11, x31 // x31 = &B[g]
ld x5, 0(x30) // f = A[f]
addi x12, x30, 8
ld x30, 0(x12)
add x30, x30, x5
sd x30, 0(x31)
```



# MFLOPS (*million of floating-point operations per second*)

- Basado en operaciones y no en instrucciones.

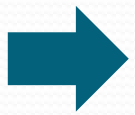
$$MFLOPS = \frac{\text{Operaciones de coma flotante realizadas}}{T_{EJEC} \times 10^6}$$

Inconvenientes:

- No todas las operaciones de coma flotante tienen la misma complejidad  $\Rightarrow$  MFLOPS normalizados: Cada operación se multiplica por un peso que es proporcional a su complejidad.

Ejemplo de asignación de pesos:

- ADD, SUB, COMPARE, MULT  $\Rightarrow$  1 operación normalizada
- DIVIDE, SQRT  $\Rightarrow$  4 operaciones normalizadas
- EXP, SIN, ATAN, ...  $\Rightarrow$  8 operaciones normalizadas
- El formato de los números en coma flotante puede variar de una arquitectura a otra y, por tanto, los resultados de las operaciones podrían tener diferente exactitud. Además, ¿y si no necesito las operaciones en coma flotante en mi servidor?

 **Conclusión final:** Tampoco nos vale y no hay más candidatos. Nos contentaremos con el tiempo de ejecución ( $T_{EJEC}$ ) de un determinado programa o conjunto de programas  $\rightarrow$  El índice de rendimiento va a depender de la carga con la que se haga el test.

# La carga real

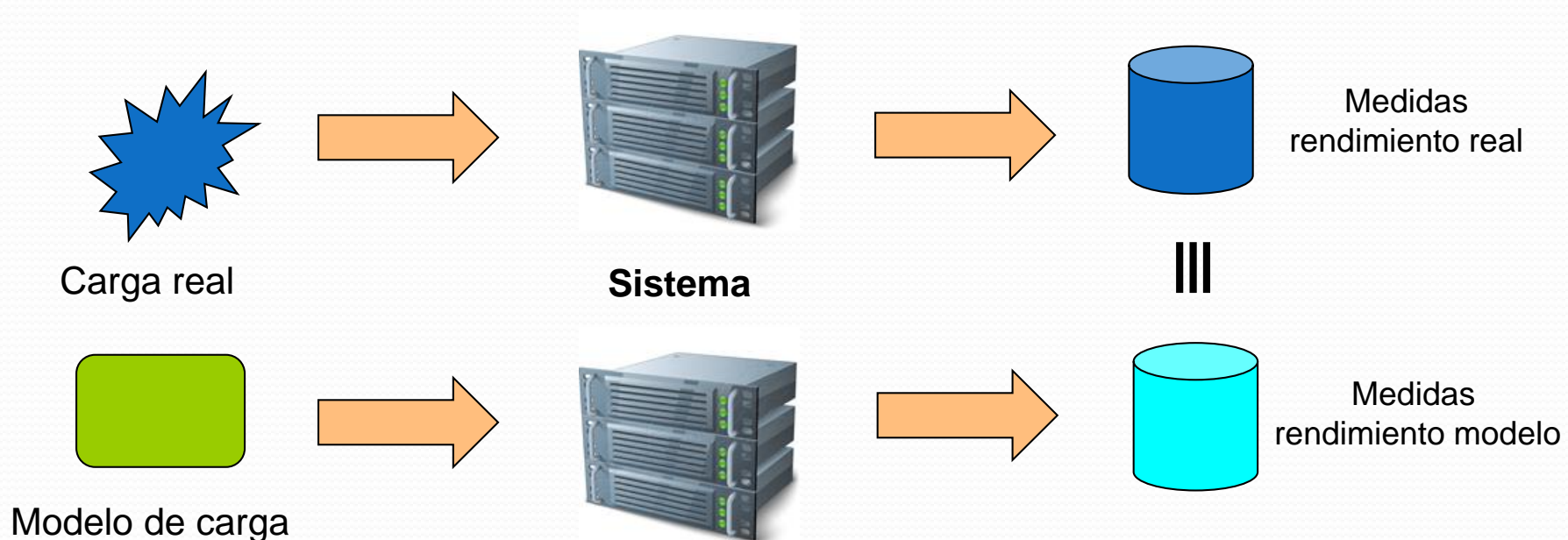
- Difícil de utilizar en la evaluación de sistemas.
  - Varía a lo largo del tiempo.
  - Resulta complicado reproducirla.
  - Interacciona con el sistema informático.



- Es más conveniente utilizar un **modelo** de la carga real como carga de prueba (test workload) para hacer comparaciones.

# Representatividad del modelo de carga

- Los modelos de carga son aproximaciones que representan una abstracción de la carga que recibe un sistema informático. El modelo de la carga:
  - Debe ser lo más representativo posible de la carga real.
  - Debe ser lo más simple/compacto que sea posible (tiempos de medición y espacio en memoria razonables).



# Principales estrategias para obtener modelos de carga

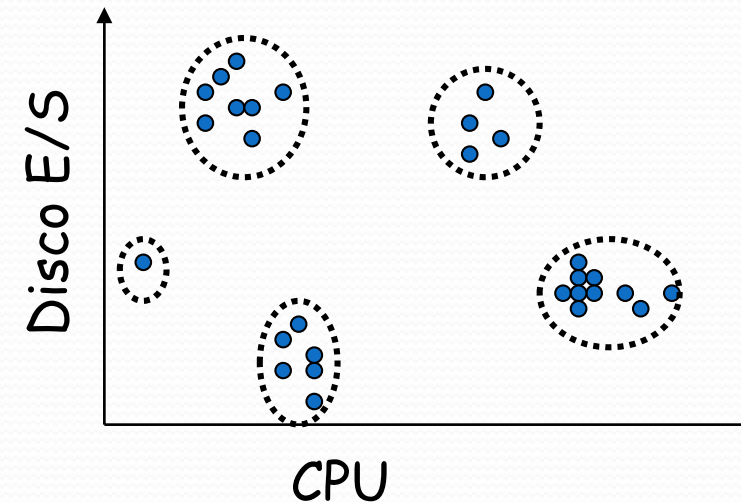
- Ajustar un modelo paramétrico “**personalizado**” a partir de la monitorización del sistema ante la carga real (*caracterización de la carga*).



- Usar programas de prueba que usen un modelo **genérico** de carga lo más similar posible al que se quiere reproducir (*referenciación o benchmarking*).

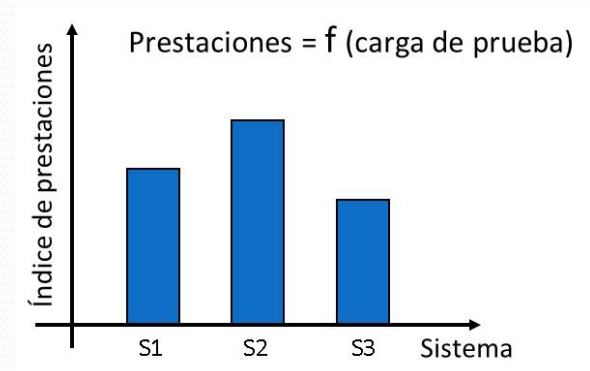
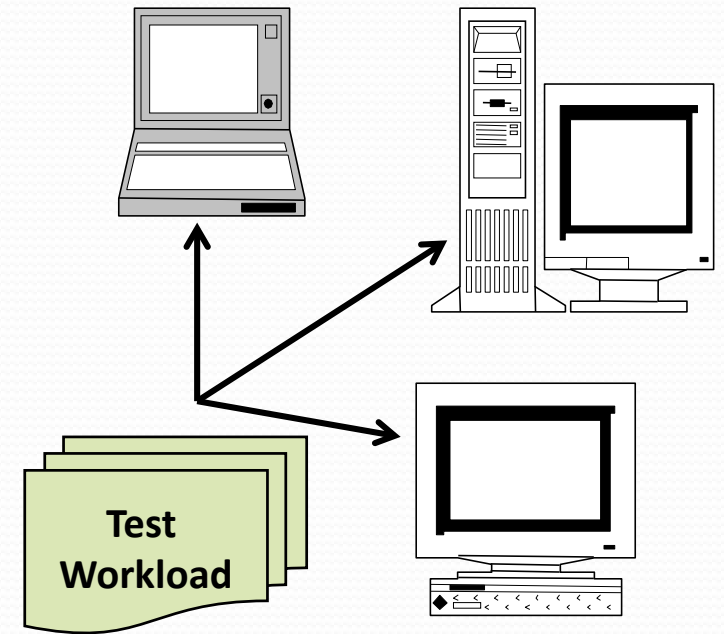
# Caracterización de la carga

- La forma más fácil para obtener un modelo de la carga que debe realizar un servidor durante un determinado periodo de tiempo consiste en:
  - Identificar los recursos que más demande la carga (CPU, memoria, discos, red, etc.)
  - Elegir los parámetros característicos de dichos recursos (utilización de CPU, lecturas/escrituras que hay que hacer en cada disco, lecturas/escrituras a memoria, número de accesos a la red, etc.)
  - Medir el valor de dichos parámetros usando monitores de actividad (muestreo).
  - Analizar los datos: medias, histogramas, agrupamiento o *clustering*, etc.
  - Generar el modelo de carga seleccionando *representantes de la carga* (=solicitudes al servidor) junto con información estadística sobre su distribución temporal.



# Referenciación (*Benchmarking*)

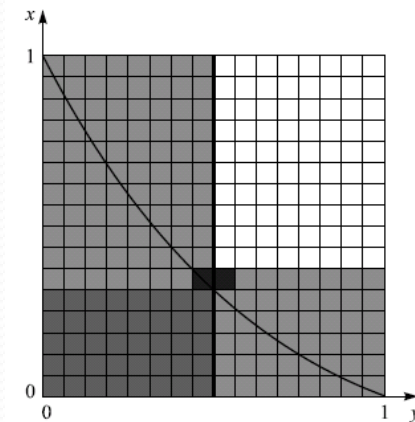
- Consiste en utilizar un programa o un conjunto de programas (*benchmark programs*) con el fin de comparar alguna característica del rendimiento entre equipos informáticos. Hay dos características principales que definen a un *benchmark*:
  - La **carga de prueba** (*test workload*) específica con la que estresa el sistema evaluado.
  - El conjunto de reglas que se deben seguir para la correcta ejecución, obtención y validación de los resultados.





# Tipos de programas de benchmark: según la estrategia de medida

- Programas que miden el tiempo necesario para ejecutar una cantidad pre-establecida de tareas.
  - La mayoría de benchmarks.
- Programas que miden la cantidad de tareas ejecutadas para un tiempo de cómputo pre-establecido.
  - SLALOM: Mide la exactitud de la solución de un determinado problema que se puede alcanzar en 1 minuto de ejecución.
- Programas que permiten variar tanto la cantidad de tareas como el tiempo de cómputo para adaptarlos a cada sistema.
  - HINT: Calcula los límites inferior y superior de una integral hasta que el sistema se quede sin recursos.



# Tipos de programas de benchmark: según la generalidad del test

- Microbenchmarks o benchmarks para **componentes**: estresan componentes o agrupaciones de componentes concretos del sistema: procesador, caché, memoria, discos, red, procesador+caché, procesador+compilador+memoria virtual, etc.
- Macrobenchmarks o benchmarks de sistema **completo** o de **aplicación real**: la carga intenta imitar situaciones reales (normalmente servidores con muchos clientes) típicas de algún área. P.ej. e-comercio, servidores web, servidores de ficheros, servidores de bases de datos, sistemas de ayuda a la decisión, paquetes ofimáticos + correo electrónico + navegación, etc.

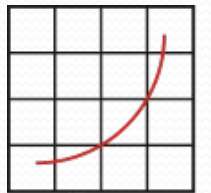


# Ejemplos de microbenchmarks

- Whetstone (1976)
  - Mide el rendimiento de las operaciones en coma flotante por medio de pequeñas aplicaciones científicas que usan sumas, multiplicaciones y funciones trigonométricas.
- Linpack (1983)
  - Mide el rendimiento de las operaciones en coma flotante a través de un algoritmo para resolver un sistema denso de ecuaciones lineales. El benchmark incorpora una rutina para comprobar que la solución a la que se llega es la correcta con un grado de exactitud prefijado.
- Dhrystone (1984)
  - Mide el rendimiento de operaciones con enteros, esencialmente por medio de operaciones de copia y comparación de cadenas de caracteres.

# Ejemplos de microbenchmarks (II)

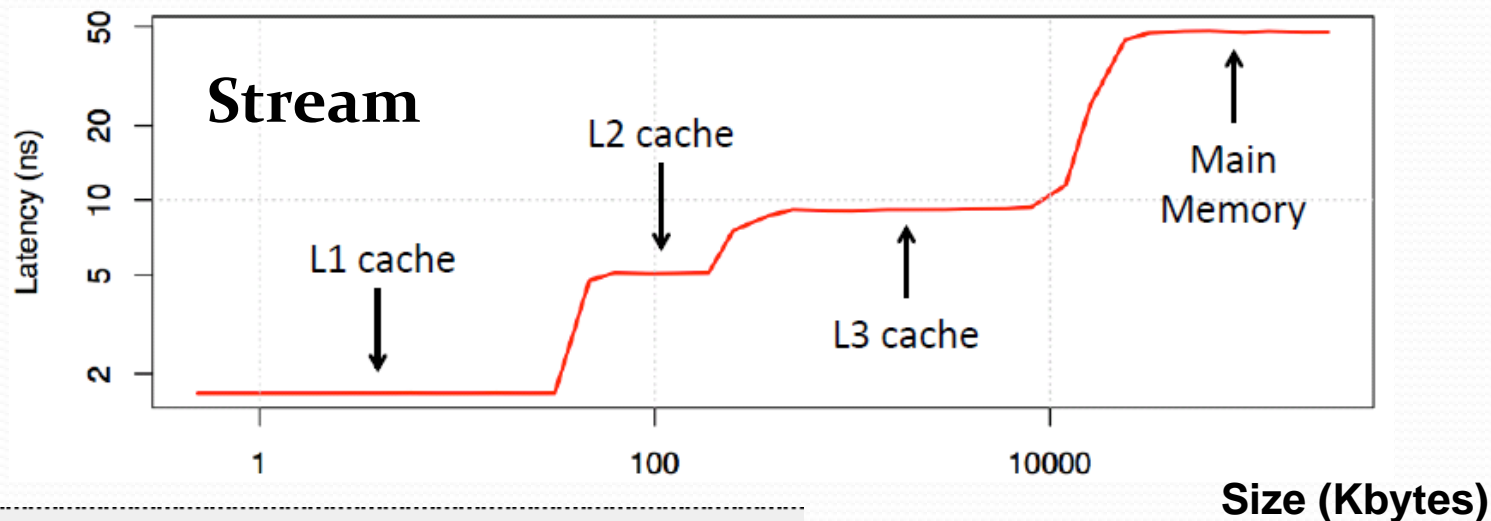
- Stream: para medir el ancho de banda de la memoria <https://github.com/jeffhammond/STREAM>.
- IOzone: rendimiento del sistema de ficheros (lecturas y escrituras a/desde el disco duro), <http://www.iozone.org/>. Igualmente HD Tune (Windows, <http://www.hdtune.com/>), Iometer (<http://www.iometer.org/>), fio (flexible I/O tester, Linux) o el comando 'hdparm -tT' (Linux).
- Netperf: rendimiento TCP y UDP (Linux y Windows). Se usa en combinación con otro programa (netserver) que debe estar instalado en el servidor. <http://www.netperf.org/netperf/>. También pchar (=traceroute que calcula el ancho de banda por cada salto).
- También existen aplicaciones que incorporan varios **paquetes de microbenchmarks** para poder realizar diversos tests de forma cómoda:
  - Lmbench (Unix, <http://lmbench.sourceforge.net>).
  - Phoronix Test Suite (Open Source, <https://www.phoronix-test-suite.com/>).
  - AIDA64 (Windows, <http://www.aida64.com>).
  - Sandra (Windows, <http://www.sisoftware.net>).
  - SPEC (<http://www.spec.org>).



spec<sup>®</sup>



# Ejemplos de micro-benchmarks (III)



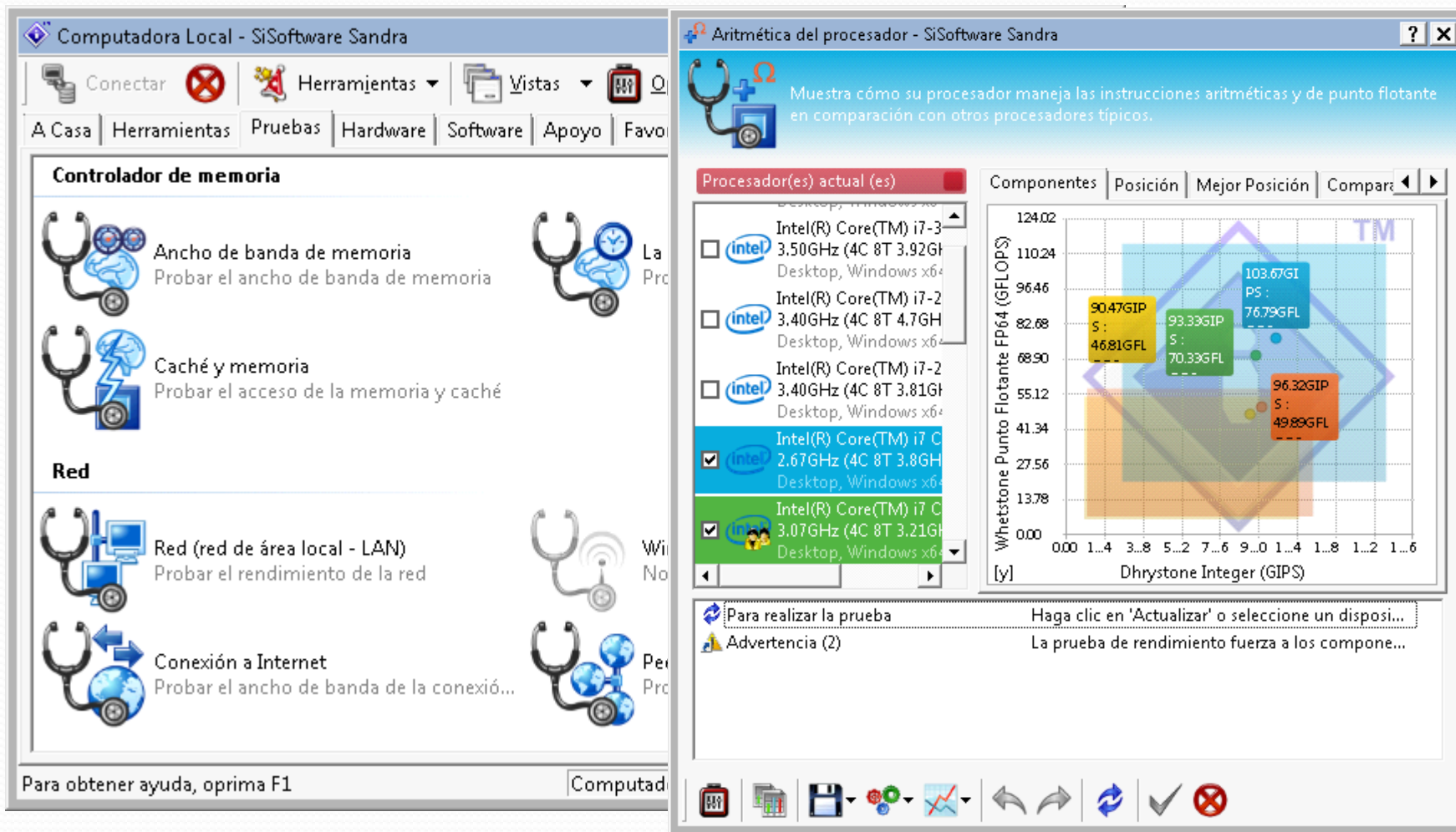
```
$ fio --name=seqwrite --rw=write --bs=128k --size=122374m
[...]
```

seqwrite: (groupid=0, jobs=1): err= 0: pid=22321  
write: io=122374MB, bw=840951KB/s, iops=6569 , runt=149011msec  
clat (usec): min=41 , max=133186 , avg=148.26, stdev=1287.17  
lat (usec): min=44 , max=133188 , avg=151.11, stdev=1287.21  
bw (KB/s) : min=10746, max=1983488, per=100.18%, avg=842503.94,  
stdev=262774.35  
cpu : usr=2.67%, sys=43.46%, ctx=14284, majf=1, minf=24  
IO depths : 1=100.0%, 2=0.0%, 4=0.0%, 8=0.0%, 16=0.0%, 32=0.0%, >=64=0.0%  
submit : 0=0.0%, 4=100.0%, 8=0.0%, 16=0.0%, 32=0.0%, 64=0.0%, >=64=0.0%  
complete : 0=0.0%, 4=100.0%, 8=0.0%, 16=0.0%, 32=0.0%, 64=0.0%, >=64=0.0%  
issued r/w/d: total=0/978992/0, short=0/0/0  
lat (usec): 50=0.02%, 100=98.30%, 250=1.06%, 500=0.01%, 750=0.01%  
lat (usec): 1000=0.01%  
lat (msec): 2=0.01%, 4=0.01%, 10=0.25%, 20=0.29%, 50=0.06%  
lat (msec): 100=0.01%, 250=0.01%

fio  
Flexible I/O tester (Linux)  
<https://fio.readthedocs.io>



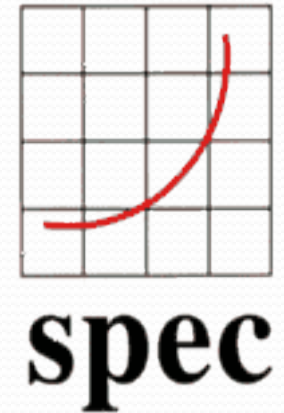
# Paquetes de microbenchmarks: SiSoftware Sandra





# SPEC (Standard Performance Evaluation Corporation)

- Es una corporación sin ánimo de lucro cuyo propósito es establecer, mantener y respaldar la estandarización de benchmarks y herramientas para evaluar el rendimiento y la eficiencia energética de los equipos informáticos.
- Miembros de la corporación (<https://www.spec.org/consortium/>): Acer Inc., Action S.A., Advanced Micro Devices, Amazon Web Services, Inc., Apple Inc., ARM, ASUSTek Computer Inc., AuriStor Inc., Bull SAS, Chengdu Haiguang IC Design Co., Ltd, Cisco Systems, Inc., Dell, Inc., Digital Ocean, Epsilon Sp. z o.o. Sp. Komandytowa, Format Sp. z o.o., Fujitsu, Gartner, Inc., Giga-Byte Technology Co., Ltd., Google Inc., Hitachi Ltd., Hitachi Vantara, HP Inc., Hewlett Packard Enterprise, IBM, Inspur Corporation, Intel, iXsystems Inc., Lenovo, Marvell Technology, Microsoft, NEC Corporation, NetApp, New H3C Technologies Co., Ltd., NVIDIA, Oracle, Principled Technologies, Pure Storage, Qualcomm Technologies Inc., Quanta Computer Inc., Red Hat, Samsung, Super Micro Computer, Inc., SUSE, Taobao (China) Software Co. Ltd., VIA Technologies, VMware, WekaIO.



*“An ounce of honest data  
is worth a pound  
of marketing hype”*

# El paquete de microbenchmarks SPEC CPU 2017

- Compuesto por cuatro conjuntos de benchmarks distintos (<http://www.spec.org/cpu2017/>):
  - SPEC`speed`®2017 Integer (rendimiento en aritmética entera)
  - SPEC`speed`®2017 Floating Point (rendimiento en coma flotante)
  - SPEC`rate`®2017 Integer (rendimiento en aritmética entera)
  - SPEC`rate`®2017 Floating Point (rendimiento en coma flotante)
    - Speed: cuánto tarda en ejecutarse un programa (tiempo de respuesta).
    - Rate: cuántos programas puedo ejecutar por unidad de tiempo (productividad).
- ¿Qué componentes se evalúan?
  - Procesador (enteros o coma flotante según el caso).
  - Sistema de memoria.
  - Compilador (C, Fortran y C++).
- Reglas estrictas para validar los resultados: <https://www.spec.org/cpu2017/Docs/runrules.html>

# El paquete de microbenchmarks SPEC CPU 2017

- SPEC CPU2017 se distribuye como una imagen ISO que contiene:
  - Código fuente de todos los programas de benchmark.
  - Data sets que necesitan algunos benchmarks para su ejecución.
  - Herramientas varias para compilación, ejecución, obtención de resultados, validación y generación de informes.
  - Documentación, incluyendo reglas de ejecución y de generación de informes.
- El tiempo de ejecución depende del índice a obtener, la máquina en la que se ejecuta y cuántas copias o subprocesos se eligen.

Metric	Config Tested	Individual benchmarks	Full Run (Reportable)
-----	-----	-----	-----
SPECrate2017_int_base	1 copy	6 to 10 minutes	2.5 hours
SPECrate2017_fp_base	1 copy	5 to 36 minutes	4.8 hours
SPECspeed2017_int_base	4 threads	6 to 15 minutes	3.1 hours
SPECspeed2017_fp_base	16 threads	6 to 75 minutes	4.7 hours

# Programas dentro de SPEC CPU 2017

- Criterios generales:
  - Han de ser aplicaciones reales.
  - Portabilidad a muchas arquitecturas: Intel y AMD x86 & x86-64, Sun SPARC, IBM POWER e IA-64.
- Ejemplo: SPECspeed<sup>®</sup>2017 Integer: 10 programas (la mayoría en C y C++)
  - 600.perlbench\_s      Intérprete de Perl
  - 657.xz\_s              Utilidad de compresión
  - 602.gcc\_s             Compilador de C
  - 623.xalancbmk\_s      Conversión XML a HTML
  - ...
- Ejemplo: SPECspeed<sup>®</sup>2017 Floating Point: 10 programas (la mayoría en Fortran y C)
  - 619.lbm\_s             Dinámica de fluidos
  - 621.wrf\_s             Predicción meteorológica
  - 638.imagick\_s        Procesamiento de imágenes
  - ...

# Índices de prestaciones en SPEC CPU2017

- Índices de prestaciones (índices SPEC)
  - Aritmética entera: CPU2017IntegerSpeed\_**peak**, CPU2017IntegerSpeed\_**base**, CPU2017IntegerRate\_**peak**, CPU2017IntegerRate\_**base**.
  - Aritmética en coma flotante: CPU2017FP\_Speed\_**peak**, CPU2017FP\_Speed\_**base**, CPU2017FP\_Rate\_**peak**, CPU2017FP\_Rate\_**base**.
- Significado de “base” y “peak”:
  - Base: Compilación en modo conservador: todos los programas escritos en el mismo lenguaje usan las mismas opciones de compilación.
  - Peak: Rendimiento pico, permitiendo que cada uno escoja las opciones de compilación óptimas para cada programa.
- Cálculo
  - Cada programa del benchmark se ejecuta 3 veces y se escoge el resultado intermedio (se descartan los 2 extremos). El índice final es la media geométrica de las ganancias en velocidad con respecto a una máquina de referencia (Sun Fire V490 con procesador UltraSPARC IV+).
- Ejemplo:

$$\text{SPEC\_CPU2017IntegerSpeed}_{base} = \sqrt[10]{\frac{t_1^{REF}}{t_1^{base}} \times \frac{t_2^{REF}}{t_2^{base}} \times \dots \times \frac{t_{10}^{REF}}{t_{10}^{base}}}$$

# Resultados de SPEC CPU2017IntegerSpeed



## All SPEC CPU2017 Integer Speed Results Published by SPEC

These results have been submitted to SPEC; see [the disclaimer](#) before studying any results.

[Search published CPU2017 results](#)

Last update: 2017-10-19 11:49

### CPU2017 Integer Speed (7):

[Search in CPU2017 Integer Speed results](#)

Test Sponsor	System Name	Parallel	Base Threads	Processor			Results	
				Enabled Cores	Enabled Chips	Threads/ Core	Base	Peak
HPE	Integrity Superdome X (384 core, 2.20 GHz, Intel Xeon E7-8890 v4) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	No	384	384	16	2	5.31	5.86
HPE	ProLiant DL580 Gen9 (2.20 GHz, Intel Xeon E7-8890 v4) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	No	96	96	4	1	5.35	5.95
HPE	ProLiant ML350 Gen9 (2.20 GHz, Intel Xeon E5-2699 v4) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	No	44	44	2	1	5.80	6.43
HPE	ProLiant DL380 Gen10 (2.10 GHz, Intel Xeon Platinum 8170) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	Yes	52	52	2	1	8.96	Not Run
HPE	ProLiant DL380 Gen10 (2.10 GHz, Intel Xeon Platinum 8176) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	Yes	56	56	2	1	9.16	Not Run
Huawei	Huawei 2288H V5 (Intel Xeon Platinum 8180) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	Yes	56	56	2	1	9.46	9.79
Oracle Corporation	Sun Fire V490 <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	Yes	1	8	4	1	1.00	Not Run



# Resultados de SPEC CPU2017IntegerSpeed (II)

Hardware		Software	
<b>CPU Name:</b>	Intel Xeon E7-8890 v4	<b>OS:</b>	SUSE Linux Enterprise Server 12 (x86_64) SP1
<b>Max MHz:</b>	3400		3.12.53-60.30-default
<b>Nominal:</b>	2200	<b>Compiler:</b>	C/C++: Version 17.0.0.098 of Intel C/C++
<b>Enabled:</b>	384 cores, 16 chips, 2 threads/core		Compiler for Linux;
<b>Orderable:</b>	2 to 16 chips		Fortran: Version 17.0.0.098 of Intel Fortran
<b>Cache L1:</b>	32 KB I + 32 KB D on chip per core		Compiler for Linux
<b>L2:</b>	256 KB I+D on chip per core	<b>Parallel:</b>	No
<b>L3:</b>	60 MB I+D on chip per chip	<b>Firmware:</b>	HP Bundle: 008.004.084 SFW: 043.025.000 08/16/2016
<b>Other:</b>	None	<b>File System:</b>	xfs
<b>Memory:</b>	4 TB (128 x 32 GB 2Rx4 PC4-2400T-L, running at 1600 MHz)	<b>System State:</b>	Run level 5 (multi-user, w/GUI)
<b>Storage:</b>	8 x C8S59A, 900 GB 10 K RPM SAS	<b>Base Pointers:</b>	64-bit
<b>Other:</b>	None	<b>Peak Pointers:</b>	32/64-bit
		<b>Other:</b>	Microquill SmartHeap V10.2

**Results Table**

Benchmark	Base							Peak						
	Threads	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio	Threads	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio
600.perlbench_s	384	365	4.86	<b>358</b>	<b>4.96</b>	357	4.98	384	298	5.95	295	6.02	<b>295</b>	<b>6.01</b>
602.gcc_s	384	553	7.20	<b>546</b>	<b>7.29</b>	546	7.29	384	540	7.37	<b>535</b>	<b>7.45</b>	534	7.45
605.mcf_s	384	<b>866</b>	<b>5.45</b>	866	5.45	898	5.26	384	708	6.67	<b>700</b>	<b>6.75</b>	699	6.75
620.omnetpp_s	384	<b>276</b>	<b>5.90</b>	271	6.03	289	5.65	384	251	6.50	<b>247</b>	<b>6.61</b>	246	6.64
623.xalancbmk_s	384	189	7.50	<b>188</b>	<b>7.52</b>	187	7.57	384	<b>179</b>	<b>7.91</b>	179	7.93	180	7.87
625.x264_s	384	<b>283</b>	<b>6.24</b>	282	6.25	283	6.23	384	<b>271</b>	<b>6.51</b>	272	6.49	270	6.52
631.deepsjeng_s	384	<b>407</b>	<b>3.52</b>	408	3.52	407	3.52	384	343	4.18	343	4.18	<b>343</b>	<b>4.18</b>
641.leela_s	384	460	3.64	460	3.63	<b>460</b>	<b>3.63</b>	384	438	3.90	<b>430</b>	<b>3.88</b>	440	3.88

# Resultados de SPEC CPU2017IntegerSpeed (III)

## Base Optimization Flags

### C benchmarks:

`-qopt-prefetch -qopt-mem-layout-trans=3 -DSPEC_SUPPRESS_OPENMP`

### C++ benchmarks:

`-Wl,-z,muldefs -qopt-prefetch -qopt-mem-layout-trans=3 -DSPEC_SUPPRESS_OPENMP -L/sh10.2 -lsmartheap64`

## Peak Optimization Flags

### C benchmarks:

600.perlbench\_s: `-prof-gen(pass 1) -prof-use(pass 2) -O2 -xCORE-AVX2 -auto-p32 -ipo -qopt-prefetch -O3 -no-prec-div  
-qopt-mem-layout-trans=3 -DSPEC_SUPPRESS_OPENMP`

602.gcc\_s: Same as 600.perlbench\_s

605.mcf\_s: `-prof-gen(pass 1) -prof-use(pass 2) -ipo -xCORE-AVX2 -O3 -no-prec-div -qopt-prefetch -qopt-mem-layout-trans=3  
-DSPEC_SUPPRESS_OPENMP`

625.x264\_s: Same as 600.perlbench\_s

657.xz\_s: Same as 600.perlbench\_s

### C++ benchmarks:

620.omnetpp\_s: `-Wl,-z,muldefs -prof-gen(pass 1) -prof-use(pass 2) -ipo -xCORE-AVX2 -O3 -no-prec-div -auto-p32 -qopt-prefetch  
-qopt-mem-layout-trans=3 -DSPEC_SUPPRESS_OPENMP -L/sh10.2 -lsmartheap64`

623.xalancbmk\_s: Same as 620.omnetpp\_s

631.deepsjeng\_s: `-Wl,-z,muldefs -prof-gen(pass 1) -prof-use(pass 2) -ipo -xCORE-AVX2 -O3 -no-prec-div -qopt-prefetch  
-qopt-mem-layout-trans=3 -DSPEC_SUPPRESS_OPENMP -L/sh10.2 -lsmartheap64`

641.leela\_s: Same as 620.omnetpp\_s

## Ejemplo de cálculo de SPEC CPU2017IntegerSpeed<sub>base</sub>

Benchmark	t <sup>REF</sup> (s)	Exp1 (s)	Exp2 (s)	Exp3 (s)	t <sup>base</sup> (s)	t <sup>REF</sup> / t <sup>base</sup>
<a href="#">600.perlbench_s</a>	1774	365	<a href="#">358</a>	357	358	4,96
<a href="#">602.gcc_s</a>	3981	553	<a href="#">546</a>	546	546	7,29
<a href="#">605.mcf_s</a>	4721	<a href="#">866</a>	866	898	866	5,45
<a href="#">620.omnetpp_s</a>	1630	<a href="#">276</a>	271	289	276	5,91
<a href="#">623.xalancbmk_s</a>	1417	189	<a href="#">188</a>	187	188	7,54
<a href="#">625.x264_s</a>	1764	<a href="#">283</a>	282	283	283	6,23
<a href="#">631.deepsjeng_s</a>	1432	<a href="#">407</a>	408	407	407	3,52
<a href="#">641.leela_s</a>	1706	469	469	<a href="#">469</a>	469	3,64
<a href="#">648.exchange2_s</a>	2939	<a href="#">329</a>	329	329	329	8,93
<a href="#">657.xz_s</a>	6182	2165	2161	<a href="#">2164</a>	2164	2,86

$$\text{SPEC\_CPU2017IntegerSpeed}_{base} = \sqrt[10]{\frac{t_1^{REF}}{t_1^{base}} \times \frac{t_2^{REF}}{t_2^{base}} \times \dots \times \frac{t_{10}^{REF}}{t_{10}^{base}}} = \sqrt[10]{4,96 \times 7,29 \times 5,45 \times \dots} = 5,31$$

# Benchmarks de sistema completo: TPC

- TPC (*Transactions Processing Performance Council*, <http://www.tpc.org>): Organización sin ánimo de lucro especializada en benchmarks relacionados con comercio electrónico y con bases de datos.

The screenshot shows the TPC website with a navigation menu on the left and a main content area. The navigation menu includes links to Home, About the TPC, Benchmarks, Enterprise BMs (TPC-C, TPC-DI, TPC-DS, TPC-E, TPC-H, TPC-VMS), Express BMs (TPCx-BB, TPCx-HCI, TPCx-HS, TPCx-IoT, TPCx-V), Common Specifications (TPC-Pricing, TPC-Energy), and Obsolete BMs (TPC-A). The main content area features a header with the TPC logo and a tagline, followed by a navigation bar with Document Search and Member Login. The main section is titled 'TPC Benchmarks & Benchmark Results' and contains a table of active benchmarks. The table has two columns: the benchmark name and a link to the results. The benchmarks listed are Transaction Processing - OLTP (TPC-C, TPC-E), Decision Support (TPC-H, TPC-DS, TPC-DI), and Virtualization (TPC-VMS, TPCx-V). To the right of the table, there are several links: About the TPC, TPC - Spreadsheets of TPC Results, TPC-E - Top Ten Performance Results, TPC - Who We Are, TPC-C - All Results - Sorted by Performance, TPC-Tools Download TPC-H, and TPC-Tools Download - Thank You TPC-H.

**TPC™**  
developing data-centric benchmark standards and disseminating objective, verifiable performance data to the industry... The TPC is a

[Document Search](#) [Member Login](#)

[Home](#)  
[About the TPC](#)  
[Benchmarks](#)  
**Enterprise BMs**  
TPC-C  
TPC-DI  
TPC-DS  
TPC-E  
TPC-H  
TPC-VMS  
**Express BMs**  
TPCx-BB  
TPCx-HCI  
TPCx-HS  
TPCx-IoT  
TPCx-V  
**Common Specifications**  
TPC-Pricing  
TPC-Energy  
**Obsolete BMs**  
TPC-A

**TPC Benchmarks & Benchmark Results**  
Please select any of the active TPC benchmarks below. All available options will be displayed.

Transaction Processing - OLTP	<a href="#">TPC-C</a>
	<a href="#">TPC-E</a>
Decision Support	<a href="#">TPC-H</a>
	<a href="#">TPC-DS</a>
	<a href="#">TPC-DI</a>
Virtualization	<a href="#">TPC-VMS</a>
	<a href="#">TPCx-V</a>

[About the TPC](#)  
[TPC - Spreadsheets of TPC Results](#)  
[TPC-E - Top Ten Performance Results](#)  
[TPC - Who We Are](#)  
[TPC-C - All Results - Sorted by Performance](#)  
[TPC-Tools Download TPC-H](#)  
[TPC-Tools Download - Thank You TPC-H](#)

# Benchmarks de sistema completo: TPC

- Principales benchmarks:
  - TPC-C: Tipo OLTP (*on-line transaction processing*). Simula una gran compañía con varios almacenes, cada uno con 100.000 productos y tiene 3000 clientes. Peticiones que involucran acceso a las bases de datos tanto locales como distribuidas.
  - TPC-E: Tipo OLTP. Simula una correduría de bolsa en donde hay una única base de datos central. El benchmark es escalable de modo que se pueden simular transacciones de compañías de diversos tamaños.
  - TPC-H, TPC-DS: Tipo DS (*decision support*). Se deben ejecutar consultas altamente complejas a una gran base de datos y analizar enormes volúmenes de datos (minería de datos, big-data).
- Métricas: peticiones/transacciones procesadas por unidad de tiempo (*tps/tpm/tph*) superando unos ciertos requisitos de tiempos de respuesta. También: coste por petición procesada (incluido mantenimiento) y consumo de potencia por petición procesada.

# TPC-H: Búsqueda de resultados

TPC™

disseminating objective, verifiable performance data to the industry... The TPC is a non-profit corporation focused on developing data-centric benchmarks

Document Search

Home

About the TPC

- Who We Are
- Contact Us
- Privacy Policy
- Stay Connected
- Press
- TPC FAQ
- History

Benchmarks

- Enterprise BMs
  - TPC-C
  - TPC-DS
  - TPC-E
  - TPC-H
  - TPC-VMS
- Express BMs
  - TPCx-BB
  - TPCx-HS
  - TPCx-V
- Common Specifications
  - TPC-Pricing
  - TPC-Energy
- Submission Checklist
- Obsolete BMs
  - TPC-A
  - TPC-App
  - TPC-B
  - TPC-D
  - TPC-R

TPC-H Advanced Sort Results List (V2.2 ) As of 25-Oct-2017 at 08:51 [Pacific Time Zone]

Note 1: The TPC believes that comparisons of TPC-H results measured against different database sizes are misleading and discourages such comparisons.

Note 2: The TPC believes it is not valid to compare prices or price/performance of results in different currencies.

Filter Options

Scale Factor <= 3000 Enter Numeric

Add Row

Sort Options

Availability Date Descending

Add Row

Display Options

# of results to display first 10

Display withdrawn results: none

Display Historical Results: No

☐ Specification Revision

☐ Total System Price

☐ OS Software Name

☐ Server CPU Name & Processors/Cores/Threads

☐ Cluster

☐ Include Energy Data

Color Legend for results selected:

- Results displayed on a white background are results which are either **In Review** results which have been **Accepted** by the TPC.

Show Results

TPC-H Advanced Sorting Results

32



# TPC-H: Búsqueda de resultados

TPC-H Advanced Sorting Results

Sponsor	System	Scale Factor	Performance (QphH)	Price/QphH	System Availability	Date Submitted	DB Software Name
 Hewlett Packard Enterprise	<a href="#">HPE ProLiant DL380 Gen9</a>	1,000	717,101	0.61 USD	10/19/2017	4/17/2017	Microsoft SQL Server 2017 Enterprise Edition
 Hewlett Packard Enterprise	<a href="#">HPE ProLiant DL380 Gen9</a>	1,000	543,102	0.69 USD	7/31/2016	3/9/2016	Microsoft SQL Server 2016 Enterprise Edition
 Lenovo	<a href="#">Lenovo System x3850 X6</a>	3,000	969,504	0.72 USD	7/31/2016	3/9/2016	Microsoft SQL Server 2016 Enterprise Edition
 Hewlett Packard Enterprise	<a href="#">HPE ProLiant DL380 Gen9</a>	1,000	678,492	0.64 USD	7/31/2016	3/24/2016	Microsoft SQL Server 2016 Enterprise Edition
 Hewlett Packard Enterprise	<a href="#">HPE ProLiant DL580 Gen9 Action Vector 5.0</a>	3,000	2,140,307	0.38 USD	7/31/2016	6/2/2016	Action Vector 5.0
 CISCO	<a href="#">Cisco UCS C460 M4 Server</a>	3,000	1,071,018	0.60 USD	6/1/2016	5/14/2016	Microsoft SQL Server 2016 Enterprise Edition
 CISCO	<a href="#">Cisco UCS C460 M4 Server</a>	3,000	725,686	1.08 USD	7/14/2015	7/13/2015	Microsoft SQL Server 2014 Enterprise Edition
 Lenovo	<a href="#">Lenovo System x3850 X6</a>	3,000	700,392	0.99 USD	5/26/2015	5/1/2015	Microsoft SQL Server 2014 Enterprise Edition

# TPC-H: Búsqueda de resultados

## TPC-H Result Highlights

As of 25-Oct-2017 at 3:54 PM [GMT]



### Cisco UCS C460 M4 Server

Reference URL: <http://www.tpc.org/3322>

#### Benchmark Stats

Result ID:	116051401
Status:	Accepted Result
Report Date:	05/14/16
TPC-H Rev:	2.17.1

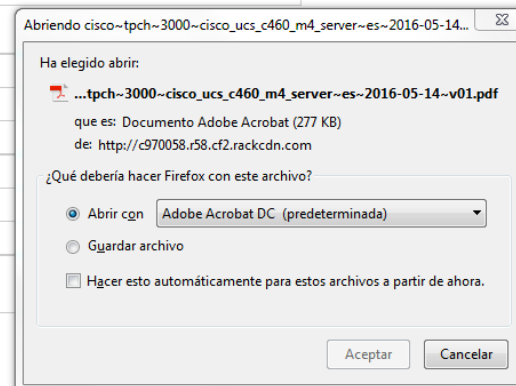
#### System Information

Total System Cost:	634,322 USD
Performance	1,071,018 QphH@3000GB
Price/Performance	.60 USD per QphH@3000GB
TPC-Energy Metric	Not reported
Availability Date	06/01/16
Database Manager	Microsoft SQL Server 2016 Enterprise Edition
Operating System	Microsoft Windows Server 2012 R2 Standard Edition


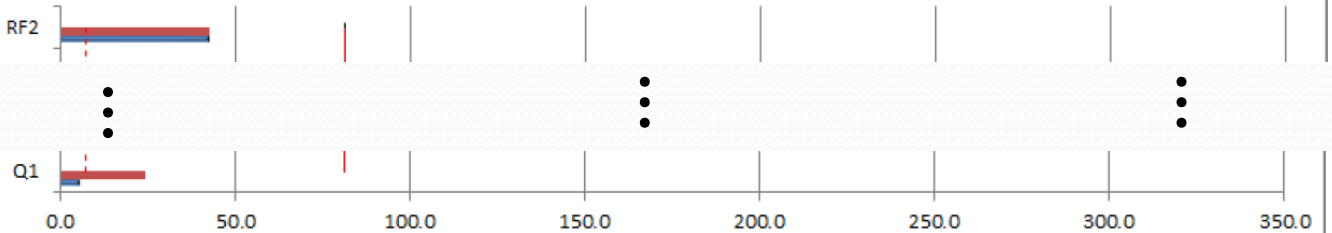
#### Server Specific Information

CPU Type:	Intel Xeon E7-8890 v3 2.50GHz
Total # of Processors:	4
Total # of Cores:	72
Total # of Threads:	144
Cluster:	No
Load Time (hours):	1.94
Total Storage/Database Size Ratio:	2.99

- Executive Summary
- Full Disclosure Report
- Supporting Files-1



# TPC-H: Búsqueda de resultados

		Cisco UCS C460 M4 Server		TPC-H Rev. 2.17.1 TPC-Pricing Rev. 2.0.0	
				Report Date: 16-May-2016	
Total System Cost		Composite Query per Hour Metric		Price / Performance	
\$634,322 USD		1,071,018.2 QphH@3000GB		\$0.60 USD \$/ QphH@3000GB	
Database Size	Database Manager	Operating System		Other Software	Availability Date
3000GB	Microsoft SQL Server 2016 Enterprise Edition	Windows 2012 R2 Standard Edition			1-June-2016
					
Database Load Time = 1h 56m 26s		Storage Redundancy Level			
Load Includes Backup: Y		Base Tables and Auxiliary Data Structures		0	
Total Data Storage / Database Size = 2.99		DBMS Temporary Space		0	
Percentage Memory / Database Size = 102.4%		OS and DBMS Software		1	
<b>System Configuration:</b> Cisco UCS C460 M4 Server Processors/Cores/Threads/Model: 4/72/144 Intel Xeon E7-8890 v3 Processor (2.5 GHz, 45MB cache, 165W) Memory: 3 TB Storage: 8 X 400GB 2.5 inch Ent Performance 12G SAS SSD (10X endurance) 4 X UCS Rack PCIe Storage 1600 GB SanDisk SX350 Medium Endurance Table Storage: 9.38 TB					

# TPC-H: Búsqueda de resultados

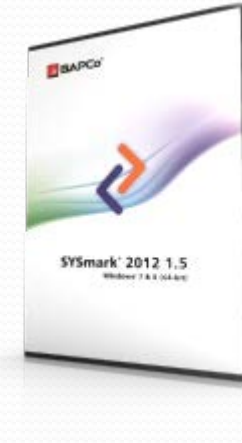
Description	Unit Price	Qty	Extended Price
<b>Server Hardware</b>			
UCS C460 M4 base chassis w/o CPU/DIMM/HDD	16,500.00	1	\$16,500.00
3YR SNTC 24X7X4OS UCS C460 M4 Server	3,487.00	1	
2.5GHz E7-8890 v3/165W/18C/45M Cache	21,000.00	4	\$84,000.00
32GB DDR4-2133-MHz RDIMM/PC4-17000/dual rank/x4/1.2v	1,100.00	96	\$105,600.00
UCS C460 M4 DDR4 Memory Riser with 12 DIMM slots	800.00	8	\$6,400.00
Riser card with 5 PCIe slots	500.00	2	\$1,000.00
400GB 2.5 inch Ent Performance 12G SAS SSD (10X endurance)	5,267.00	8	\$42,136.00
1400W V2 AC Power Supply (200 - 240V) 2U & 4U C Series	800.00	4	\$3,200.00
Power Cord, 200/240V 6A North America	0.00	4	\$0.00
Full Height PCIe slot filler for C Series	0.00	6	\$0.00
Bracket and Supercap cable for C460 M4 and 12 drive RAID	0.00	1	\$0.00
CPU Heat Sink for UCS C460 M4 Rack Server	0.00	4	\$0.00
Rail Kit for UCS C460 M4	0.00	1	\$0.00
Cisco 12G SAS Modular Raid Controller (12 port)	1,688.00	1	\$1,688.00
Cisco 12Gbps SAS 1GB FBWC Cache module (Raid 0/1/5/6)	1,217.00	1	\$1,217.00
Cisco ONE Data Center Compute Opt Out Option	0.00	1	\$0.00
UCS 2.5 inch HDD blanking panel	0.00	4	\$0.00
UCS Rack PCIe Storage 1600GB SanDisk SX350 Medium Endurance	18,133.00	4	\$72,532.00
Cisco R42610 standard rack, w/side panels	3,429.00	1	\$3,429.00
IOGEARGKM513 Spill Proof Keyboard & Mouse Combo	15.91	1	\$15.91
ASUS 19.5" VS207D-P Widescreen LED 1600x900 VGA	87.71	1	\$87.71
			<u>\$337,806</u>

# Benchmarks de sistema completo: SPEC

- **File Server: SFS2014:** Tiempos de respuesta y productividades de servidores de ficheros.
- **High Performance Computing, OpenMP, MPI, OpenCL**
  - **SPEC MPI2007:** Message Passing Interface (MPI).
  - **SPEC OMP2012:** Open MultiProcessing (OpenMP).
  - **SPEC ACCEL:** OpenCL y OpenACC
- **JAVA Cliente/Servidor**
  - **SPECjEnterprise2010:** Java Enterprise Edition (JEE).
  - **SPECjms2007:** Java Message Service (JMS).
  - **SPECjvm2008:** Java Runtime Environment (JRE).
- **Virtualization: SPECvirt\_sc2010** (Virtualización en Centros de Procesamiento de Datos).
- **Cloud: SPEC Cloud\_IaaS 2016** (Servicios en la nube)
- **Consumo de potencia: SPECpower\_ssj2008** (Rendimiento de un servidor ejecutando aplicaciones JAVA frente al consumo de potencia).

# Benchmarks de sistema completo: SYSMark 2012

- Para comparar PC con S.O. Windows.
- Considera la carga en 6 escenarios:
  - Office Productivity: Word, PowerPoint, Outlook, Acrobat ...
  - Media Creation: Adobe Photoshop, Adobe Premiere...
  - Web Development: Dreamweaver, IE, Firefox...
  - Data/Financial Analysis: Excel.
  - 3D Modeling: Autodesk 3ds Max, AutoCAD, Google SketchUp...
  - System Management: Winzip, Firefox installer.
- Con cada programa se ejecuta un conjunto de tareas de acuerdo con un modelo de comportamiento de un usuario “habilitado”.
- El tiempo medio de ejecución de los benchmarks de cada categoría se normaliza (ratio) respecto de una máquina de referencia. Finalmente, el índice SYSMark2012 se calcula mediante la media geométrica de los ratios obtenidos.





## 4.2. Análisis de los resultados de un test de rendimiento

# ¿Cómo expresar el resultado final tras la ejecución de un test de rendimiento?

- El rendimiento es una variable multidimensional.
  - Habría de expresarse mediante múltiples índices.
  - Sin embargo, las comparaciones son más sencillas si se usa un único índice de rendimiento (a minimizar o maximizar).
- ¿Cómo concentrar todos los índices en uno solo?
  - Utilizar la *mejor* variable que represente el rendimiento.
  - Método habitual de síntesis: uso de algún tipo de **media**.



# La media aritmética

- Dado un conjunto de  $n$  medidas,  $t_1, \dots, t_n$ , definimos su media aritmética:

$$\bar{t} = \frac{1}{n} \sum_{k=1}^n t_k$$

- Si no todas las medidas tienen la misma importancia, se puede asociar a cada medida  $t_k$  un peso  $w_k$ , obteniéndose la **media aritmética ponderada**:

$$\overline{t_W} = \sum_{k=1}^n w_k \times t_k$$

$$\text{con } \sum_{k=1}^n w_k = 1$$

Si  $t_k$  es el tiempo de ejecución del programa de benchmark  $k$ -ésimo en la máquina a testar,  $w_k$  podría escogerse, por ejemplo, inversamente proporcional a dicho tiempo de ejecución en una determinada máquina de referencia:

$$w_k \equiv \frac{C}{t_k^{REF}}$$



$$C = \frac{1}{\sum_{k=1}^n 1/t_k^{REF}}$$

# La media geométrica

- Dado un conjunto de  $n$  medidas,  $S_1, \dots, S_n$ , definimos su media geométrica:

$$\overline{S_g} = \sqrt[n]{\prod_{k=1}^n S_k} = \left( \prod_{k=1}^n S_k \right)^{1/n}$$

- Propiedad: cuando las medidas son ganancias en velocidad (*speedups*) con respecto a una máquina de referencia, este índice mantiene el mismo orden en las comparaciones independientemente de la máquina de referencia elegida. Usado en los benchmarks de SPEC y SYSMARK.

$$SPEC(M) = \sqrt[n]{\frac{t_1^{REF}}{t_1^M} \times \frac{t_2^{REF}}{t_2^M} \times \dots \times \frac{t_n^{REF}}{t_n^M}} = \frac{\sqrt[n]{t_1^{REF} \times t_2^{REF} \times \dots \times t_n^{REF}}}{\sqrt[n]{t_1^M \times t_2^M \times \dots \times t_n^M}}$$

$$SPEC(M1) > SPEC(M2) \Leftrightarrow \sqrt[n]{t_1^{M1} \times t_2^{M1} \times \dots \times t_n^{M1}} < \sqrt[n]{t_1^{M2} \times t_2^{M2} \times \dots \times t_n^{M2}}$$

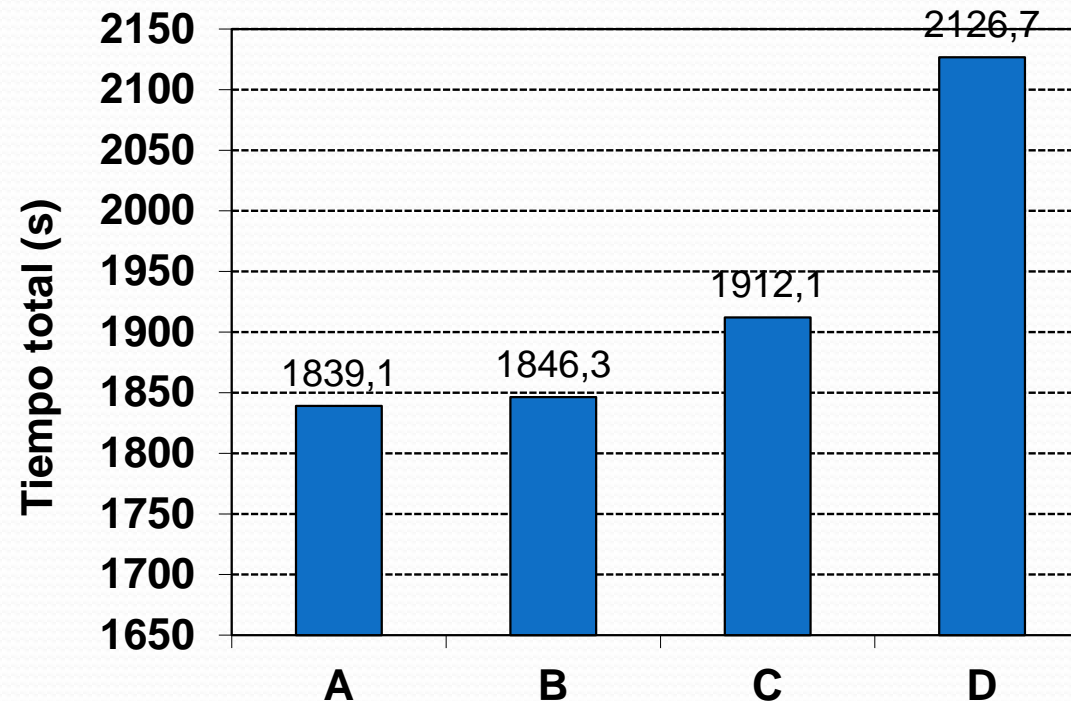
# Ejemplo de comparación con tiempos

Programa	$t^{\text{REF}}$ (s)	$t^A$ (s)	$t^B$ (s)	$t^C$ (s)	$t^D$ (s)
1	1400	141	170	136	134
2	1400	154	166	215	25
3	1100	96,8	94,2	146	201
4	1800	271	283	428	523
5	1000	83,8	90,1	77,4	81,2
6	1200	179	189	199	245
7	1300	120	131	87,7	75,5
8	300	151	158	138	192
9	1100	93,5	122	88	118
10	1900	133	173	118	142
11	1500	173	170	179	240
12	3000	243	100	100	150
Suma	17000	1839,1	1846,3	1912,1	2126,7

- La máquina más rápida es “A” ya que es la que tarda menos en ejecutar, uno tras otro, todos los programas del benchmark (1839,1 segundos).

# Comparación con el tiempo total

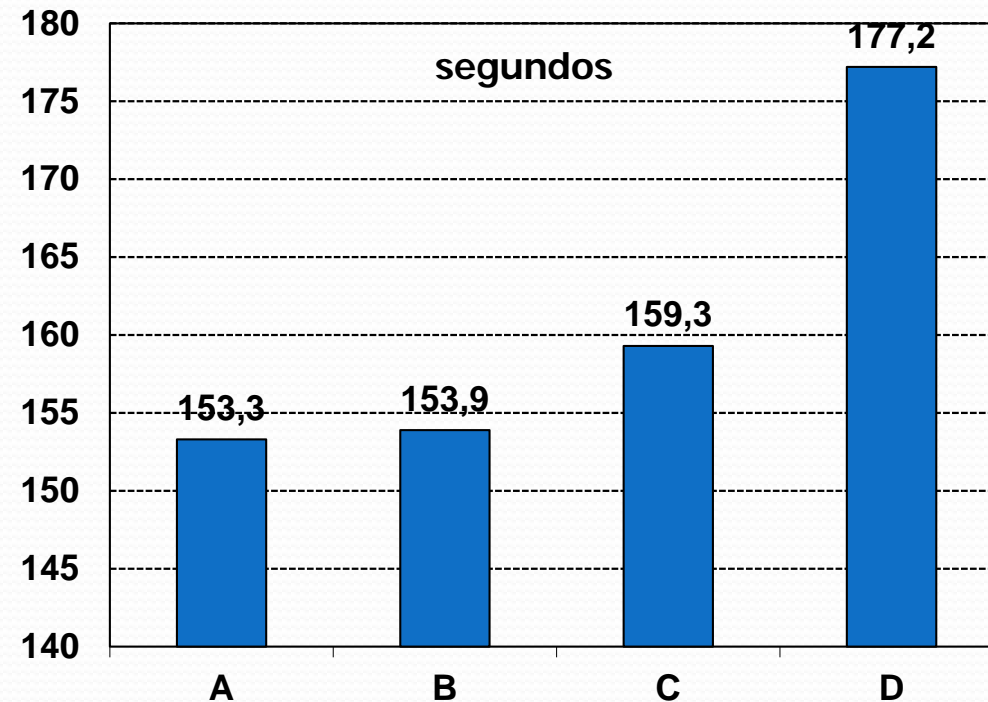
- Ordenación con el tiempo total:
  - De más rápida a más lenta: A, B, C, D
  - Esto no significa que A sea siempre la más rápida (depende del programa), aunque, en conjunto, sí que lo es.





# Comparación con la media aritmética

$$\begin{aligned}\bar{t}_A &= \frac{1}{12} \sum_{k=1}^{12} t_k^A = 153,3s \\ \bar{t}_B &= \frac{1}{12} \sum_{k=1}^{12} t_k^B = 153,9s \\ \bar{t}_C &= \frac{1}{12} \sum_{k=1}^{12} t_k^C = 159,3s \\ \bar{t}_D &= \frac{1}{12} \sum_{k=1}^{12} t_k^D = 177,2s\end{aligned}$$



- La máquina que ejecuta los programas del benchmark, uno tras otro, en menor tiempo es la de menor media aritmética de los tiempos de ejecución.

# Usando la media aritmética ponderada

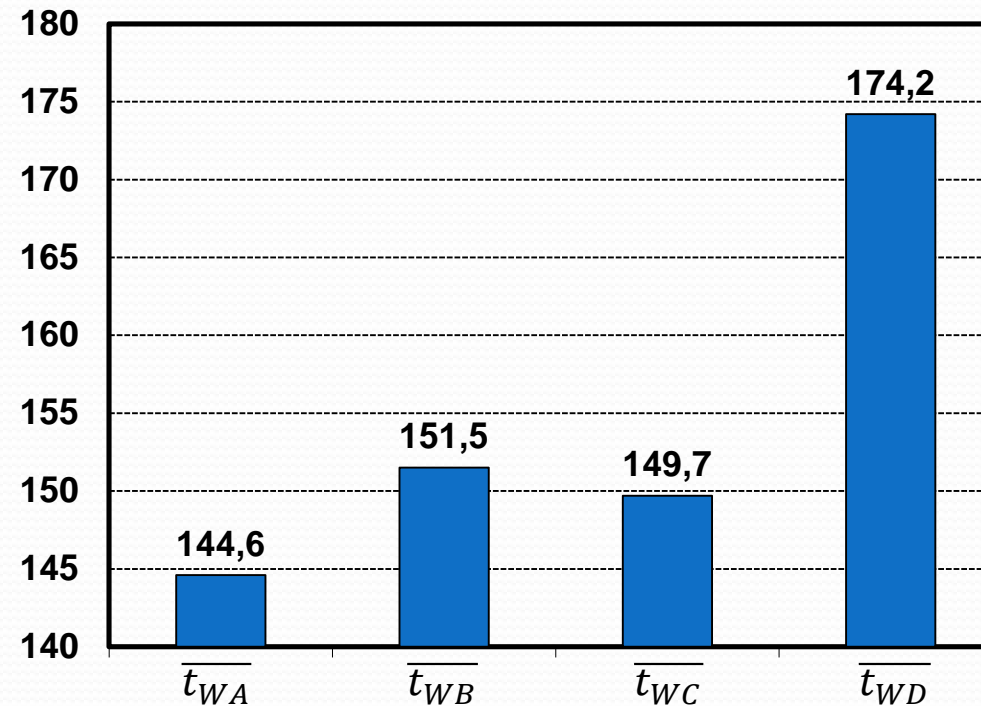
Prog	$t_k^{REF}$ (s)	$w_k$
1	1400	0,06
2	1400	0,06
3	1100	0,08
4	1800	0,05
5	1000	0,09
6	1200	0,07
7	1300	0,07
8	300	0,30
9	1100	0,08
10	1900	0,05
11	1500	0,06
12	3000	0,03
Suma	17000	1

$$w_k \equiv \frac{C}{t_k^{REF}}$$

$$C = \frac{1}{\sum_{k=1}^n 1/t_k^{REF}} = 88,77s$$

$$\overline{t_{WA}} = \sum_{k=1}^{12} w_k \times t_k^A = 144,6s$$

Igualmente, se calculan:  
 $\overline{t_{WB}}$ ,  $\overline{t_{WC}}$  y  $\overline{t_{WD}}$



- Según este criterio, la máquina “más rápida” sería la de menor tiempo medio ponderado de ejecución. Nótese que esta ponderación depende, en este ejemplo, de la máquina de referencia.

# Usando la media geométrica de *speedups*

- Calculamos la ganancia en velocidad de cada máquina con respecto a la máquina de referencia (tal y como lo hacen SPEC y Sysmark):

Programa	$t^{\text{REF}}(s)$	$s^A$ speedup	$s^B$ speedup	$s^C$ speedup	$s^D$ speedup
1	1400	9,9	8,2	10,3	10,4
2	1400	9,1	8,4	6,5	56,0
3	1100	11,4	11,7	7,5	5,5
4	1800	6,6	6,4	4,2	3,4
5	1000	11,9	11,1	12,9	12,3
6	1200	6,7	6,3	6,0	4,9
7	1300	10,8	9,9	14,8	17,2
8	300	2,0	1,9	2,2	1,6
9	1100	11,8	9,0	12,5	9,3
10	1900	14,3	11,0	16,1	13,4
11	1500	8,7	8,8	8,4	6,3
12	3000	12,3	30,0	30,0	20,0
M. Geom.		8,78	8,66	8,97	9,00

- El *speedup* es un índice a maximizar. Según este índice, la “mejor máquina” es ¡¡¡la D!!!

# ¿A quién beneficia la decisión de usar la media geométrica de *speedups*?

J8										=MEDIA.GEOM(J2:J5)	
	A	B	C	D	E	F	G	H	I	J	
	Prog. Bench.	tREF(s)	tA(s)	tB(s)	tC(s)	tD(s)	tREF/tA	tREF/tB	tREF/tC	tREF/tD	
1											
2	1	200	100	99	1	1	2,00	2,02	200,0	200,0	
3	2	200	100	101	133	1	2,00	1,98	1,50	200,0	
4	3	200	100	100	133	1	2,00	2,00	1,50	200,0	
5	4	200	100	100	133	397	2,00	2,00	1,50	0,50	
6	Suma	800	400	400	400	400					
7											
8				Media Geométrica			2,0000	2,0001	5,11	44,81	



Se premian las mejoras sustanciales. No se castigan empeoramientos no tan sustanciales. Debemos ser MUY cuidadosos con las comparaciones y saber qué estamos haciendo realmente.



# Conclusiones de este análisis

- Intentar reducir un conjunto de medidas de un benchmark a un solo “valor medio” final no es una tarea trivial.
- La media aritmética de los tiempos de ejecución de un benchmark es una medida fácilmente interpretable e independiente de ninguna máquina de referencia. El menor valor nos indica la máquina que ha ejecutado el **conjunto** de programas del benchmark, uno tras otro, en un tiempo menor.
- La media aritmética ponderada nos permite asignar más peso a algunos programas que a otros. Esa ponderación debería realizarse, idealmente, según las necesidades del usuario. Si se hace de forma dependiente de los tiempos de ejecución de una máquina de referencia, la elección de ésta puede influir significativamente en los resultados.
- La media geométrica de las ganancias en velocidad con respecto a una máquina de referencia es un índice de interpretación compleja cuya comparación no depende de la máquina de referencia. Premia mejoras sustanciales con respecto a algún programa del benchmark y no castiga al mismo nivel los empeoramientos.

## 4.3. Comparación de prestaciones en presencia de aleatoriedad

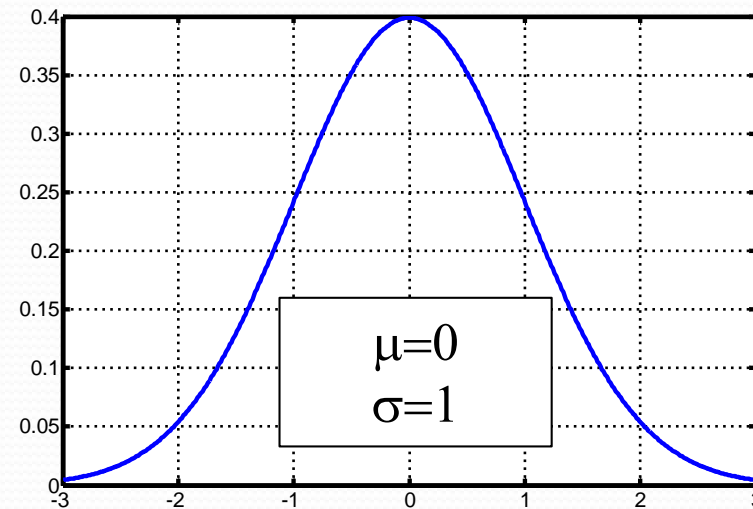


# Repaso de Estadística: Distribución Normal

- Independientemente de qué índice se escoja, un buen ingeniero debería, en primer lugar, determinar si las diferencias entre las medidas obtenidas por un test de rendimiento en presencia de aleatoriedad son **estadísticamente significativas** → Necesitaremos repasar algunos conceptos de estadística.
- **Distribución normal:** Es una distribución de probabilidad caracterizada por su media  $\mu$  y su varianza  $\sigma^2$  cuya función de probabilidad viene dada por:

$$Prob(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

La probabilidad de obtener un elemento en el rango  $[\mu - 2\sigma, \mu + 2\sigma]$  es del 95%



- Teorema del límite central: la suma de un conjunto grande de muestras aleatorias de cualquier distribución e independientes entre sí pertenece una distribución normal.

# Repaso de Estadística: Distribución t de Student

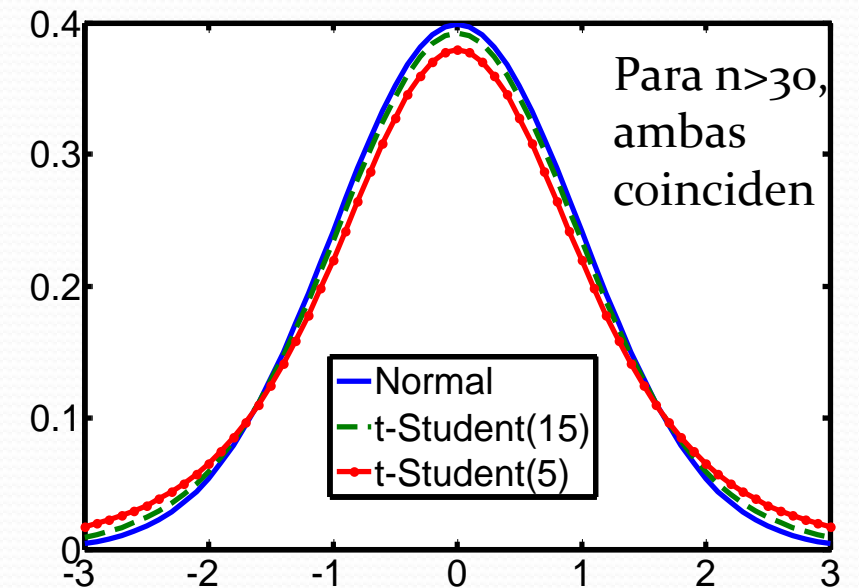
Si extraemos  $n$  muestras  $d_i$  pertenecientes a una distribución Normal de media  $\bar{d}_{real}$ , y calculo la siguiente medida (=estadístico):

$$t_{exp} = \frac{\bar{d} - \bar{d}_{real}}{s/\sqrt{n}}$$

siendo  $\bar{d}$  la media muestral y  $s$  la desviación típica muestral

$$\bar{d} = \frac{\sum_{i=1}^n d_i}{n} \quad s = \sqrt{\frac{\sum_{i=1}^n (d_i - \bar{d})^2}{n-1}} = \sqrt{\frac{\sum_{i=1}^n d_i^2 - n \cdot \bar{d}^2}{n-1}}$$

y repetimos el experimento muchas veces, veremos que esos  $t_{exp}$  se distribuyen según la distribución t-Student con  $n-1$  grados de libertad.



$$Prob(t) = \frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi} \Gamma(\frac{\nu}{2})} \left(1 + \frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

¿Para qué me puede servir esto?

# Ejemplo 1: Test de rendimiento entre A y B

- Tiempos de ejecución (en segundos) de 6 programas (P1...P6) en dos máquinas diferentes (A y B)

Programa	tA (s)	tB (s)	$d_i = tA_i - tB_i$
P1	142	100	42
P2	139	92	47
P3	152	128	24
P4	112	82	30
P5	156	148	8
P6	166	171	-5
Suma	867	721	

¿Son significativas estas diferencias?

$$\bar{d} = 24,3 \text{ seg}$$

$$s = 19,9 \text{ seg}$$

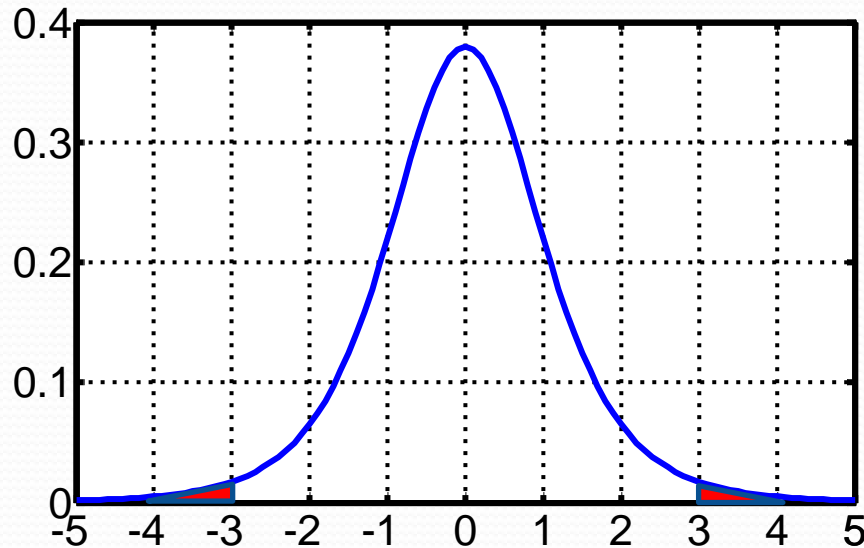
- Si partimos de la hipótesis (“hipótesis nula”,  $H_0$ ) de que las máquinas tienen rendimientos equivalentes, entonces las diferencias se deben a una suma de factores aleatorios independientes. En ese caso  $d_i$  serán muestras de una distribución normal de media cero ( $\bar{d}_{real} = 0$ ). Por tanto:

$$t_{exp} = \frac{\bar{d}}{s/\sqrt{n}} = 2,99$$

pertenece a una distribución t de Student con  $6-1=5$  grados de libertad. ¿Qué probabilidad hay de que esto sea realmente así?

# Nivel o Grado de Significatividad ( $\alpha$ )

- Distribución t de Student con 5 grados de libertad ( $T_5$ ).



$$P - value = P(|t| \geq |t_{exp}|) \text{ en } T_{n-1} \\ = 2 \times P(t \leq -|t_{exp}|) \text{ en } T_{n-1}$$

$$= \text{DISTR.T2C}(2,99;5) = 0,03 \text{ (Excel).}$$

$$= \text{DISTR.T}(2,99;5;2) = 0,03 \text{ (Calc).}$$

$$= 2 \cdot \text{tcdf}(-2.99,5) = 0,03 \text{ (Matlab).}$$

La probabilidad de obtener un valor de  $|t|$  igual o superior a 2,99 de una distribución t de Student con 5 grados de libertad es de 0,03 (**P-value** (Valor-P)= 0,03). ¿Es eso mucho o poco? Debemos definir un umbral: **nivel o grado de significatividad  $\alpha$** . Normalmente,  $\alpha=0,05$  (5%).

Conclusión: Si  $P\text{-value} < \alpha$  diremos que, para un grado de significatividad  $\alpha$  o para un **nivel de confianza**  $(1-\alpha)*100$  (normalmente 95%), las máquinas tienen rendimientos estadísticamente diferentes. En ese caso, B sería 1,2 veces más rápida que A en ejecutar el conjunto de programas ( $867/721=1,2$ ). En caso contrario, no podríamos descartar la hipótesis de que las máquinas tengan rendimientos equivalentes.

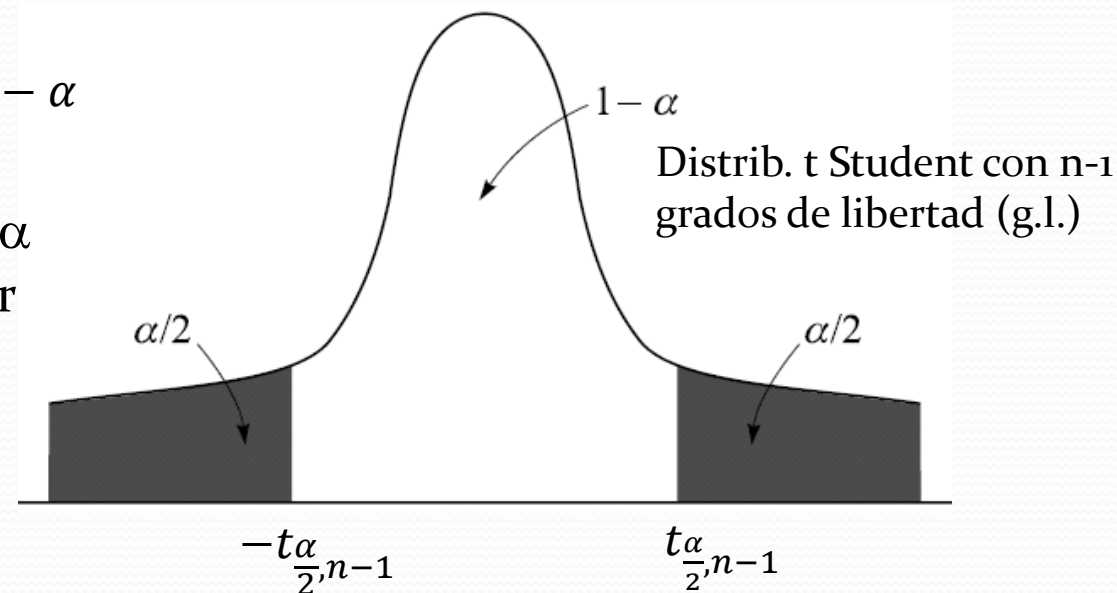
# Intervalos de confianza para $t_{\text{exp}}$

- Para un nivel de significatividad  $\alpha$  (típ.  $0,05 = 5\%$ ), buscamos el valor  $t_{\alpha/2, n-1}$  que cumpla  $Prob(|t| > t_{\alpha/2, n-1}) = \alpha$  o equivalentemente:

$$Prob(-t_{\alpha/2, n-1} \leq t \leq t_{\alpha/2, n-1}) = 1 - \alpha$$

- Diremos que para un nivel de confianza  $1-\alpha$  (típ.  $0,95 = 95\%$ ), **para aceptar  $H_0$**  el valor de  $t_{\text{exp}}$  debería situarse en el intervalo:

$$[-t_{\alpha/2, n-1}, t_{\alpha/2, n-1}]$$



- A dicho intervalo se le denomina **intervalo de confianza** de la medida para un nivel de significatividad  $\alpha$ . Teniendo en cuenta que:

$$Prob(-t_{\alpha/2, n-1} \leq t \leq t_{\alpha/2, n-1}) = 1 - 2 \times Prob(t \leq -t_{\alpha/2, n-1}) = 1 - 2 \times Prob(t > t_{\alpha/2, n-1})$$

es fácil demostrar que  $t_{\alpha/2, n-1}$  cumple que (ver figura):

$$Prob(t \leq -t_{\alpha/2, n-1}) = Prob(t > t_{\alpha/2, n-1}) = \alpha/2$$

# Intervalos de confianza para $t_{exp}$ (cont.)

- En el caso del *Ejemplo 1*, para un nivel de significatividad de  $\alpha=0,05$ , buscamos  $t_{\alpha/2, n-1}$  tal que:

$$Prob(t \leq -t_{\alpha/2, n-1}) = \alpha/2 = 0,025$$

para una distribución t de Student con 5 grados de libertad. Eso se puede obtener, por ejemplo:

- En *Excel*, haciendo:  $ABS(INV.T(\alpha/2; n-1)) = ABS(INV.T(0,025; 5)) = 2,57$ .
- En *Calc*,  $DISTR.T.INV(\alpha; n-1) = DISTR.T.INV(0,05; 5) = 2,57$ .
- En *Matlab*, haciendo:  $abs(tinv(\alpha/2, n-1)) = abs(tinv(0,025, 5)) = 2,57$ .

$$|t_{\alpha/2, n-1}|$$

- Dicho de otra manera, si las diferencias entre los tiempos de ejecución de ambas máquinas se debieran a factores aleatorios, existiría un 95% de probabilidad de que

$$t_{exp} = \frac{\bar{d}}{s/\sqrt{n}}$$

se encuentre en el rango  $[-t_{\alpha/2, n-1}, t_{\alpha/2, n-1}] = [-t_{0,025, 5}, t_{0,025, 5}] = [-2,57, 2,57]$ .



Como  $t_{exp} = 2,99$  no está en ese rango, concluiremos nuevamente que la hipótesis de que ambas máquinas tienen rendimientos equivalentes no es cierta con el 95% de confianza.



# Intervalos de confianza para $\bar{d}_{real}$

- Acabamos de ver que si las diferencias entre los tiempos de ejecución de ambas máquinas se debieran a factores aleatorios, existiría un 95% de probabilidad de que  $t_{exp}$  se encuentre en el rango  $[-t_{\frac{\alpha}{2}, n-1}, t_{\frac{\alpha}{2}, n-1}] = [-2,57, 2,57]$ .

- Como

$$t_{exp} = \frac{\bar{d} - \bar{d}_{real}}{s/\sqrt{n}} \in [-|t_{\frac{\alpha}{2}, n-1}|, |t_{\frac{\alpha}{2}, n-1}|]$$

sin más que identificar  $t_{exp}$  con los valores límite  $\pm t_{\frac{\alpha}{2}, n-1}$  sabemos que, de ser  $H_0$  cierta, habrá un 95% de probabilidad de que el valor medio real  $\bar{d}_{real}$  de las diferencias entre los tiempos de ejecución se encuentre en el intervalo:

$$\bar{d}_{real} \in \left[ \bar{d} - \frac{s}{\sqrt{n}} \times |t_{\frac{\alpha}{2}, n-1}|, \bar{d} + \frac{s}{\sqrt{n}} \times |t_{\frac{\alpha}{2}, n-1}| \right] = 24,3 \mp 20,9 = [3,4, 45,2] s$$

Y el problema se transforma simplemente en comprobar si ese valor medio real  $\bar{d}_{real}$  puede o no ser **cero**.

➡ En nuestro ejemplo, como el intervalo no incluye el cero, concluiremos una vez más que la hipótesis de que ambas máquinas pueden tener rendimientos equivalentes **no** es cierta al 95% de confianza.

# En resumen: Test t (valor-p o p-value)

- Ejecución de  $n$  programas en dos máquinas A y B.
- ¿Son significativas las diferencias obtenidas ( $d_i = tA_i - tB_i$ )? Hay que usar mecanismos estadísticos.

- Cálculo:

$$t_{exp} = \frac{\bar{d}}{s/\sqrt{n}} \quad \text{siendo } \bar{d} = \frac{\sum_{i=1}^n d_i}{n} \quad s = \sqrt{\frac{\sum_{i=1}^n (d_i - \bar{d})^2}{n-1}}$$

$$P - value = P(|t| \geq |t_{exp}|) \text{ en } T_{n-1}$$

- Concluiremos, para un nivel de confianza del  $(1-\alpha) \times 100$  % (típ. 95%) o para un nivel de significatividad de  $\alpha$  (típ. 5%):
  - Si  $P\text{-value} \geq \alpha$ , entonces no hay diferencias significativas (es posible que los valores de  $d_i$  sean aleatorios  $\rightarrow$  las dos alternativas pueden tener rendimientos equivalentes).
  - Si  $P\text{-value} < \alpha$ , entonces las alternativas presentan rendimientos significativamente diferentes. La que sea mejor dependerá del índice de rendimiento que se considere (tiempos medios, SPEC, etc.)

# Resumen: Test t (Intervalos de confianza para $t_{exp}$ )

- Ejecución de  $n$  programas en dos máquinas A y B.
- ¿Son significativas las diferencias obtenidas ( $d_i = tA_i - tB_i$ )? Hay que usar mecanismos estadísticos.

- Calculo: 
$$t_{exp} = \frac{\bar{d}}{s/\sqrt{n}} \quad \text{siendo } \bar{d} = \frac{\sum_{i=1}^n d_i}{n} \quad s = \sqrt{\frac{\sum_{i=1}^n (d_i - \bar{d})^2}{n-1}}$$

- Intervalo de confianza para  $t_{exp}$  (para un nivel de significatividad  $\alpha$  predeterminado, normalmente 0,05):

$$\left[ -t_{\frac{\alpha}{2}, n-1}, t_{\frac{\alpha}{2}, n-1} \right]$$

siendo  $t_{\alpha/2, n-1}$  el valor que hace que  $Prob(t \leq -t_{\alpha/2, n-1}) = \alpha/2$  para una distribución t de Student con  $n-1$  grados de libertad.

- Concluiremos, para un nivel de confianza del  $(1-\alpha) \times 100$  % (típ. 95%) o para un nivel de significatividad  $\alpha$  (típ. 5%):
  - Si  $t_{exp}$  está en el intervalo, entonces no hay diferencias significativas.
  - Si no lo está, entonces las alternativas presentan rendimientos significativamente diferentes.

# Resumen: Test t (Intervalos de confianza para $\bar{d}_{real}$ )

- Ejecución de  $n$  programas en dos máquinas A y B.
- ¿Son significativas las diferencias obtenidas ( $d_i = tA_i - tB_i$ )? Hay que usar mecanismos estadísticos.
- Intervalo de confianza para la media real de las diferencia  $\bar{d}_{real}$  (para un nivel de significatividad  $\alpha$  predeterminado, típ. 0,05):

$$\bar{d}_{real} \in \left[ \bar{d} - \frac{s}{\sqrt{n}} \times |t_{\frac{\alpha}{2}, n-1}|, \bar{d} + \frac{s}{\sqrt{n}} \times |t_{\frac{\alpha}{2}, n-1}| \right] \equiv \bar{d} \pm \frac{s}{\sqrt{n}} \times t_{\frac{\alpha}{2}, n-1}$$

siendo  $t_{\alpha/2, n-1}$  el valor que hace que  $Prob(t \leq -t_{\alpha/2, n-1}) = \alpha/2$  para una distribución t de Student con  $n-1$  grados de libertad.

- Concluiremos, para un nivel de confianza del  $(1-\alpha) \times 100$  % (típ. 95%) o para un nivel de significatividad  $\alpha$  (típ. 5%):
  - Si el intervalo incluye el cero, entonces no hay diferencias significativas.
  - Si no incluye el cero, entonces las alternativas presentan rendimientos significativamente diferentes.

## Ejemplo 2: Test de rendimiento entre A y B

- Tiempos de ejecución (en segundos) de 5 programas (P1...P5) para dos valores diferentes (A y B) de un parámetro del S.O.

Programa	tA (s)	tB (s)	$d_i = tA_i - tB_i$
P1	23	15	8
P2	28	22	6
P3	19	20	-1
P4	29	27	2
P5	36	39	-3
Suma	135	123	

¿Son significativas estas diferencias?

dato:  $|t_{0,025,4}| = 2,78$

$$\bar{d} = 2,4s$$

$$s = 4,6s$$

$$t_{exp} = \frac{\bar{d}}{s/\sqrt{n}} = 1,16$$

- $P - value = P(|t| \geq t_{exp}; n - 1) = P(|t| \geq 1,16; 4) = 0,31 (> 0,05)$
- Para un nivel de significatividad de  $\alpha=0,05$ :
  - Intervalo de confianza para  $t_{exp}$ :  $[-2.78, 2.78]$  (**dentro del intervalo**)
  - Intervalo para  $\overline{d_{real}}$ : (**incluye el cero**)

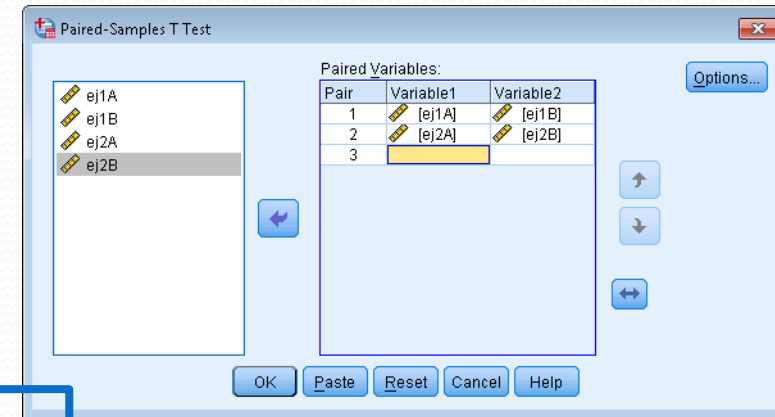
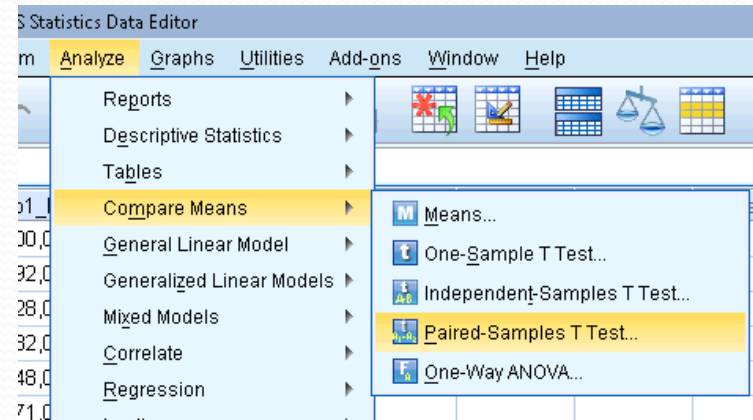
$$\bar{d} \pm \frac{s}{\sqrt{n}} \times t_{\frac{\alpha}{2}, n-1} = 2,4 \pm \frac{4,6}{\sqrt{5}} \times 2,78 = 2,4 \pm 5,72 = [-3,3, 8,1]s$$

➡ **NO** podemos descartar, al 95% de nivel de confianza, que ambos valores del parámetro del S.O. puedan tener rendimientos equivalentes.

# Test T con SPSS

\*TestT\_SPSS.sav [DataSet1] - IBM SPSS Statistics Data Editor

	ej1A	ej1B	ej2A	ej2B
1	142,00	100,00	23,00	15,00
2	139,00	92,00	28,00	22,00
3	152,00	128,00	19,00	20,00
4	112,00	82,00	29,00	27,00
5	156,00	148,00	36,00	39,00
6	166,00	171,00		
7				



Intervalo para  $\overline{d_{real}}$ :  $\bar{d} \pm \frac{s}{\sqrt{n}} \times t_{\alpha/2, n-1}$

Paired Samples Test									
		Paired Differences					$t_{exp}$ ↓ t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
					Lower	Upper			
Pair 1	ej1A - ej1B	24,33333	19,92653	8,13497	3,42172	45,24495	2,991	5	,030
Pair 2	ej2A - ej2B	2,40000	4,61519	2,06398	-3,33052	8,13052	1,163	4	,310

P-value



# Test T con Statgraphics

	ej1A	ej1B	ej2A	ej2B	Cc
1	142	100	23	15	
2	139	92	28	22	
3	152	128	19	20	
4	112	82	29	27	
5	156	148	36	39	
6	166	171			
7					
8					

## Prueba de Hipótesis para ej1A - ej1B

Prueba t

Hipótesis Nula: media = 0

Alternativa: no igual

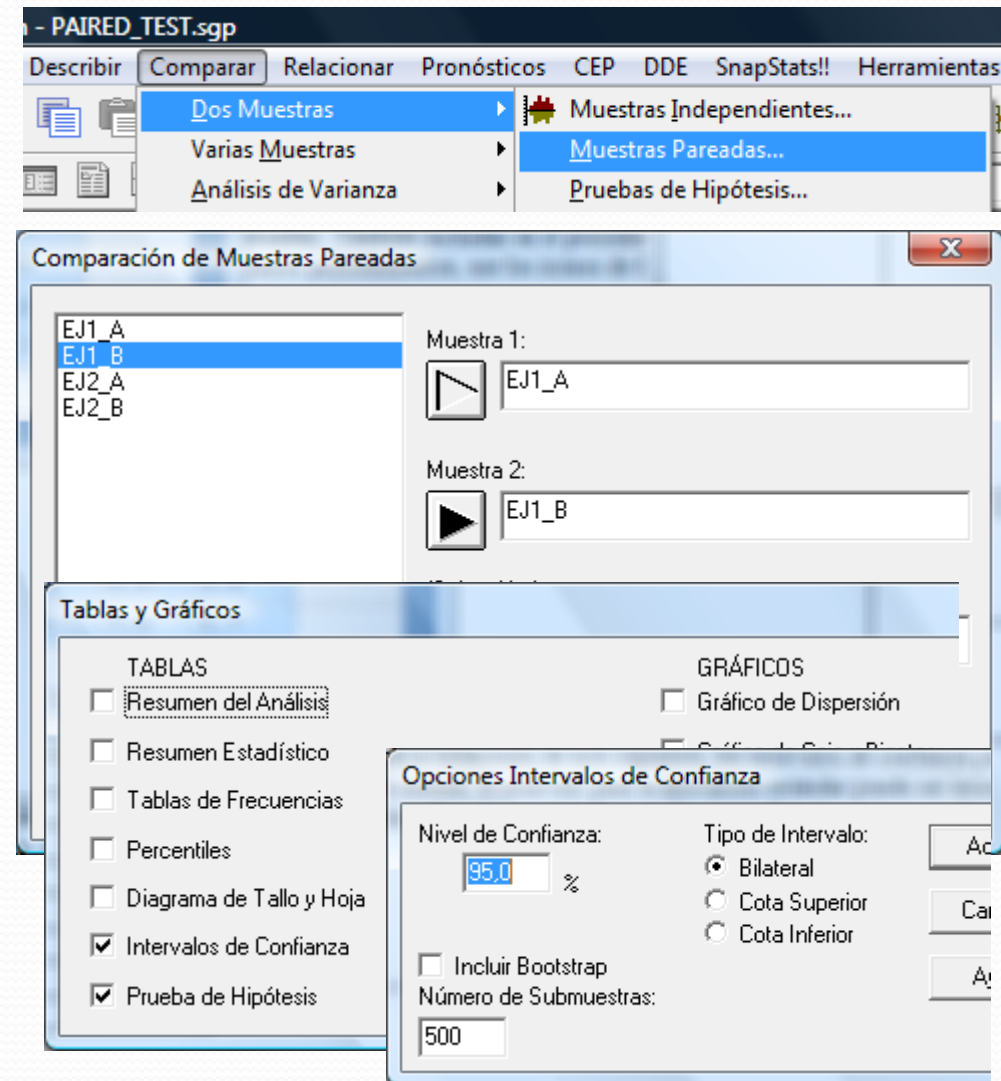
Estadístico t = 2,9912

Valor-P (P-value) = 0,0304056

Se rechaza la hipótesis nula para alfa = 0,05.

## Intervalos de Confianza para ej1A - ej1B

Intervalos de confianza del 95,0% para la media: 24,3333 +/- 20,9117 [3,42166; 45,245]



## Otra utilidad del test t: Estimación de intervalos de confianza de medias experimentales

**Hipótesis:** Realizamos  $n$  medidas  $d_i$  de un mismo fenómeno (p.ej. tiempos de ejecución de un programa, tiempos acceso de un disco duro, productividades de red,...). Si éstas pueden diferir debido a efectos aleatorios, podemos suponer que se distribuyen según una normal de media  $\bar{d}_{real}$ , que es el valor que buscamos. En ese caso, sabemos que

$$t_{exp} = \frac{\bar{d} - \bar{d}_{real}}{s/\sqrt{n}}$$

se distribuye según la distribución t-Student con  $n-1$  grados de libertad, siendo  $\bar{d}$  y  $s$  la media y la desviación típica muestrales, respectivamente.

**Por tanto**, hay un  $(1-\alpha)*100\%$  de probabilidad de que el valor medio real  $\bar{d}_{real}$  se encuentre en el intervalo:

$$\bar{d} \pm \frac{s}{\sqrt{n}} t_{\alpha/2, n-1}$$

**Utilidad:** Podemos usar esta información para determinar un intervalo de confianza para  $\bar{d}_{real}$ , y no quedarnos simplemente con el valor medio muestral.

# Ejemplo

Queremos determinar un intervalo de confianza del 95% para el tiempo medio de escritura de un determinado fichero en un disco duro. Para ello, se han realizado  $n=8$  medidas experimentales:

#exp	$t_e(\text{ms})$
1	835
2	798
3	823
4	803
5	834
6	825
7	813
8	829

$$\bar{t}_e = \frac{\sum_{i=1}^n t_{ei}}{n} = 820\text{ms} \quad s = \sqrt{\frac{\sum_{i=1}^n (t_{ei} - \bar{t}_e)^2}{n-1}} = 14\text{ms}$$

$$|t_{\alpha/2, n-1}| = |t_{0,025,7}| = 2,36$$

- En *Excel*, haciendo: ABS(INV.T(alfa/2;n-1)).
- En *Calc*, DISTR.T.INV(alfa;n-1).
- En *Matlab*, haciendo: abs(tinv(alfa/2,n-1)).

Por tanto, hay un 95% ( $\alpha=0,05$ ) de probabilidad de que el tiempo medio de escritura real de ese fichero se encuentre en el intervalo:

$$\bar{t}_e \pm \frac{s}{\sqrt{n}} t_{\alpha/2, n-1} = 820 \pm \frac{14}{\sqrt{8}} t_{\frac{0,05}{2}, 8-1} = [808, 832]\text{ms}$$

## 4.4 Diseño de experimentos de comparación de rendimiento

# Planteamiento del problema

- Supongamos que queremos determinar cuáles de los siguientes factores afectan significativamente al rendimiento de un determinado servidor:
  1. Sistema Operativo: Windows Server, CentOS, Debian, Ubuntu.
  2. Memoria RAM: 32GB, 64GB, 128GB.
  3. Discos duros: SATA, IDE, SAS.
- Y, en el caso de que afecten, cuál de los niveles del factor es significativamente mejor que el resto.
- ¿Qué experimentos debemos diseñar para ello y cómo debemos analizar los resultados?



# Terminología

- **Variable respuesta o dependiente (*métrica*):** El índice de rendimiento que usamos para las comparaciones. P.ej. tiempos de respuesta (R), productividades (X).
- **Factor:** Cada una de las *variables* que pueden afectar a la variable respuesta. P.ej. sistema operativo, tamaño de memoria, tipo de disco duro, tipo de procesador, número de microprocesadores, número de cores, tamaño de cada caché, compilador, algún parámetro configurable del S.O., etc.
- **Nivel:** Cada uno de los *valores* que puede asumir un factor. P.ej. para un S.O.: Windows, CentOS, Debian, Ubuntu; para un tipo de disco duro: SATA, IDE, SAS; para un parámetro del sistema operativo: ON, OFF, etc.
- **Interacción:** Una interacción ocurre cuando el efecto de un factor cambia para diferentes niveles de otro factor. P.ej. el hecho de usar un tipo determinado de S.O. puede afectar a cómo de importante sea usar una mayor cantidad de memoria RAM.



# Tipos de diseños experimentales

- **Diseños con un solo factor:** Se utiliza una configuración determinada como base y se estudia un factor cada vez, midiendo los resultados para cada uno de sus niveles. Problema: solo válida si descartamos que haya interacción entre factores. Número total de experimentos =  $1 + \sum_{i=1}^k (n_i - 1)$  donde  $k$  es el número de factores y  $n_i$  el número de niveles del factor  $i$ . En nuestro ejemplo, habría que hacer 8 experimentos.
  - **Diseños multi-factoriales completos:** Se prueba cada posible combinación de niveles para todos los factores. Ventaja: se analizan las interacciones entre todos los factores. Número total de experimentos =  $\prod_{i=1}^k n_i$ . En nuestro ejemplo, 36 experimentos.
  - **Diseños multi-factoriales fraccionados:** Término medio entre los anteriores. No todas las interacciones se verán reflejadas en los resultados, solo las de las interacciones que se consideren más probables.
- ☞ Todos ellos se pueden realizar con diferentes niveles de **repetición**: a) sin repeticiones, b) con todos los experimentos repetidos el mismo número de veces, c) con un número de repeticiones diferentes para cada nivel o cada factor.

# Diseños con un solo factor

- **Ejemplo:** Para el servidor principal de nuestra empresa, queremos saber si la elección del tipo de disco duro afecta al rendimiento. Para ello, se ha escogido tres tipos de discos duros: **SAS, SATA e IDE** y se ha realizado un experimento que consiste en ejecutar, en condiciones reales, un conjunto de programas usados habitualmente por el servidor y medir el **tiempo de ejecución**. Este experimento se ha repetido **5 veces**:

#Exp.	SAS (s)	SATA (s)	IDE (s)
1	103	115	143
2	97	102	134
3	123	120	139
4	106	115	135
5	116	122	129
Medias	109.0	114.8	136.0
Efectos ( $\varepsilon_j$ )	-10.9	-5.1	16.1

**$m_{\text{global}} = 119.9s$**

- ☞ ¿Tiene influencia el factor disco duro sobre el rendimiento? ¿Son las diferencias entre los discos duros significativas? **Test ANOVA.**

# Análisis de la Varianza (ANOVA) de un factor

$$\text{Modelo: } y_{ij} = m_{\text{global}} + \varepsilon_j + r_{ij} \quad i=1, \dots, n_{\text{rep}}; \quad j=1, \dots, n_{\text{niv}}$$

$y_{ij}$ : Las observaciones. En nuestro caso los tiempos de ejecución obtenidos en cada prueba. El índice  $j$  recorre los distintos niveles del factor cuya influencia se quiere medir (en nuestro caso hay  $n_{\text{niv}}=3$  niveles: SAS, SATA e IDE). El índice  $i$  recorre las distintas repeticiones para cada uno de esos niveles (en nuestro caso,  $n_{\text{rep}}=5$  repeticiones).

$m_{\text{global}}$ : Media global de todas las observaciones:

$$m_{\text{global}} = \frac{1}{n_{\text{rep}} \times n_{\text{niv}}} \sum_{i=1}^{n_{\text{rep}}} \sum_{j=1}^{n_{\text{niv}}} y_{ij}$$

$\varepsilon_j$ : Efecto debido al nivel  $j$ -ésimo:  $\varepsilon_j = \frac{1}{n_{\text{rep}}} \sum_{i=1}^{n_{\text{rep}}} y_{ij} - m_{\text{global}}$ . Se cumple que  $\sum_{j=1}^{n_{\text{niv}}} \varepsilon_j = 0$ .

$r_{ij}$ : Perturbaciones o error experimental (ruido). Deben cumplir:

- Que tengan varianza constante, independiente del nivel (si nº de exp./nivel no es homogéneo).
- Que su distribución sea normal.

Si no se cumplen, hay otros test alternativos: test de Kruskal-Wallis, test de Friedman.

☞ La principal pregunta que intenta contestar el test ANOVA es: ¿Tiene influencia el factor sobre la variable respuesta (algún  $\varepsilon_j$  es distinto de cero)?

# Análisis de la Varianza (ANOVA) de un factor (II)

El método ANOVA se basa en descomponer la varianza de las muestras en:

$$\sum_{i=1}^{n_{rep}} \sum_{j=1}^{n_{niv}} (y_{ij} - m_{global})^2 = n_{rep} \sum_{j=1}^{n_{niv}} (\varepsilon_j)^2 + \sum_{i=1}^{n_{rep}} \sum_{j=1}^{n_{niv}} (r_{ij})^2$$

Utilizando notación abreviada:

$$SST=SSA+SSE$$

- SST= Varianza total de las muestras. (Sum-of-Squares Total)
- SSA= Varianza explicada por los efectos o alternativas (intergrupos). (Sum-of-Squares Alternatives)
- SSE= Varianza residual o del error (intragrupos) (Sum-of-Squares Error)

El objetivo es contrastar la hipótesis de que el factor no influye sobre los resultados ( $\varepsilon_j \approx 0 \forall j = 1 \cdots n_{niv}$ ). Si esto es cierto, resulta que el resultado de hacer:

$$F_{exp} \equiv \frac{SSA/(n_{niv} - 1)}{SSE/(n_{niv} \times (n_{rep} - 1))} \sim F_{n_{niv}-1, n_{niv} \times (n_{rep}-1)}$$

debería ser una muestra de una distribución  $F$  de Snedecor con  $n_{niv}-1$  grados de libertad en el numerador y  $n_{niv} \times (n_{rep}-1)$  en el denominador.

# Análisis de la Varianza (ANOVA) de un factor (III)

En nuestro ejemplo:

$$SST = 2809$$

$$SSA = 2020$$

$$SSE = 789$$

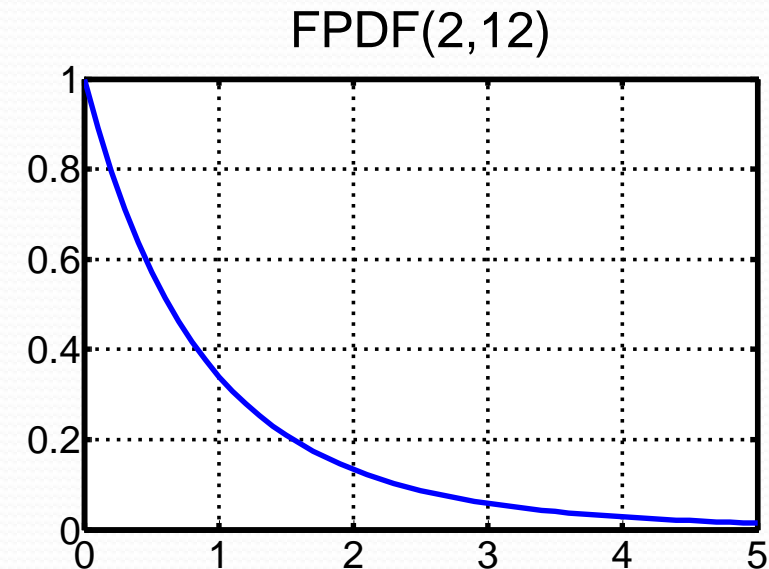
$$F_{exp} \equiv \frac{\frac{SSA}{n_{niv} - 1}}{\frac{SSE}{(n_{niv} \times (n_{rep} - 1))}} = \frac{\frac{2020}{3 - 1}}{\frac{789}{(3 \times (5 - 1))}} = 15,37$$

¿Qué probabilidad hay de que la muestra 15,37 o superior se haya extraído de una distribución  $F_{2,12}$ ?  
 $P - value = P(F \geq 15,37; 2, 12) = 0,00049$ . Identifico ese valor como la probabilidad de que la hipótesis de que el factor no influye pueda ser cierta.

*Excel y Calc:* DISTR.F(15,37;2;12); *Matlab:* 1-fcdf(15.37,2,12).

Si la probabilidad es menor que  $\alpha = 0,05$  diremos que descartamos la hipótesis de que el factor no influya a un  $(1-\alpha) \times 100\% = 95\%$  de confianza.

Si el factor influye, a continuación (análisis *post-hoc*) comparamos las medias de cada nivel unas con otras usando un *test t*: prueba de múltiples rangos o de comparaciones múltiples.

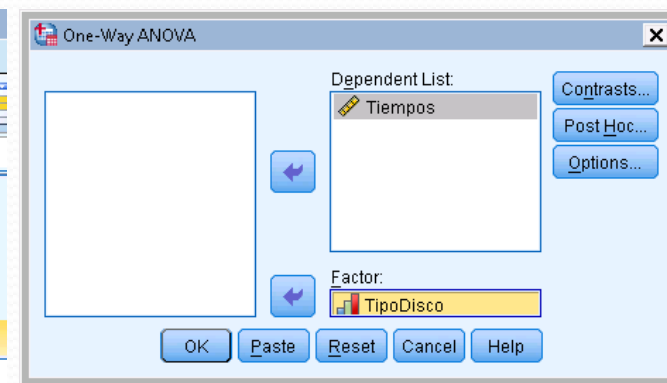
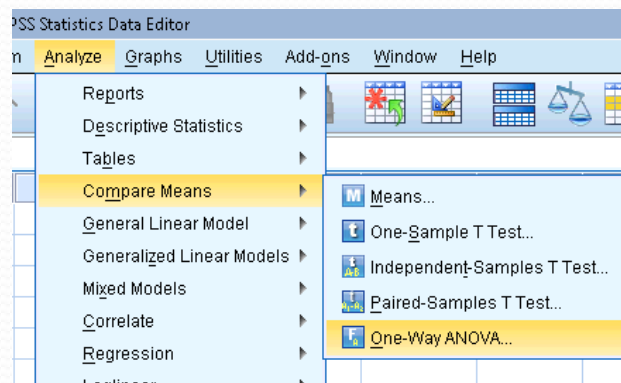


# Diseños con un solo factor con SPSS

1: SAS; 2: SATA; 3: IDE



	TipoDisco	Tiempos
1	1	103,0
2	1	97,0
3	1	123,0
4	1	106,0
5	1	116,0
6	2	115,0
7	2	102,0
8	2	120,0
9	2	115,0
10	2	122,0
11	3	143,0
12	3	134,0
13	3	139,0
14	3	135,0
15	3	129,0
16		
17		



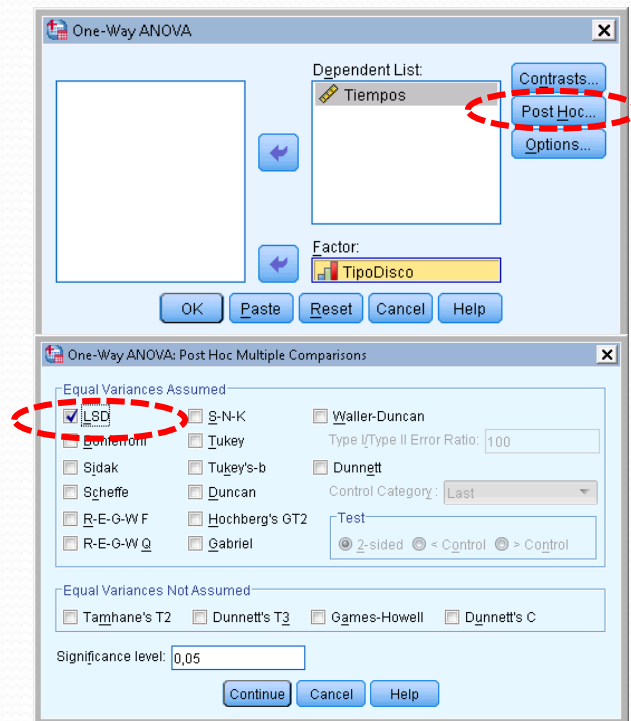
ANOVA					
Dependent Variable: Tiempos					
	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	2020,133	2	1010,067	15,366	,000
Within Groups	788,800	12	65,733		
Total	2808,933	14			

Esto demuestra que el tipo de disco duro afecta significativamente al rendimiento del equipo casi para cualquier nivel de significatividad que usemos.



# Diseños con un solo factor con SPSS (II)

Como el factor influye, hacemos ahora un test t entre cada combinación de niveles para comparar el efecto en el rendimiento de cada tipo de disco duro: **prueba de múltiples rangos o de comparaciones múltiples.**



1: SAS; 2: SATA; 3: IDE

Multiple Comparisons						
Dependent Variable Tiempos						
Test		LSD				
		Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
(I) Grupo	(J) Grupo				Lower Bound	Upper Bound
1	2	-5,8000	5,1277	,280	-16,972	5,372
	3	-27,0000*	5,1277	,000	-38,172	-15,828
2	1	5,8000	5,1277	,280	-5,372	16,972
	3	-21,2000*	5,1277	,001	-32,372	-10,028
3	1	27,0000*	5,1277	,000	15,828	38,172
	2	21,2000*	5,1277	,001	10,028	32,372

\*. The mean difference is significant at the 0.05 level.

Concluimos que, al 95% de confianza, el disco IDE es claramente peor que los otros dos, pero que las diferencias entre SAS y SATA, para este problema, no son estadísticamente significativas, por lo que podríamos decidirnos por el más barato (o hacer más pruebas para estar más seguros).

# Diseños con un solo factor con Statgraphics

The screenshot shows the Statgraphics software interface. On the left is a data table with 16 rows and 3 columns: 'TipoDiscos', 'Tiempo', and an unlabeled column. The data is as follows:

	TipoDiscos	Tiempo
1	SAS	103,0
2	SAS	97,0
3	SAS	123,0
4	SAS	106,0
5	SAS	116,0
6	SATA	115,0
7	SATA	102,0
8	SATA	120,0
9	SATA	115,0
10	SATA	122,0
11	IDE	143,0
12	IDE	134,0
13	IDE	139,0
14	IDE	135,0
15	IDE	129,0
16		
17		

In the center, the 'ANOVA Simple' dialog box is open. The 'Variable Dependiente' is 'Tiempo' and the 'Factor' is 'TipoDiscos'. The 'Tablas y Gráficos' section is also visible, showing options for 'Resumen del Análisis', 'Resumen Estadístico', 'Tabla ANOVA' (checked), 'Gráfico de Dispersión', 'ANOVA Gráfico', and 'Gráfico de Medias'.

Below the dialog box, the 'Tabla ANOVA para Tiempos por Tipo Disco' is displayed:

Fuente	Suma de Cuadrados	Gl	Cuadrado Medio	Razón-F	Valor-P
Entre grupos	2020,13	2	1010,07	15,37	0,0005
Intra grupos	788,8	12	65,7333		
Total (Corr.)	2808,93	14			

# Diseños con un solo factor con Statgraphics (II)

**Tablas y Gráficos**

<b>TABLAS</b>	<b>GRÁFICOS</b>	<b>Aceptar</b>
<input type="checkbox"/> Resumen del Análisis	<input type="checkbox"/> Gráfico de Dispersión	<b>Cancelar</b>
<input type="checkbox"/> Resumen Estadístico	<input type="checkbox"/> <b>ANOVA Gráfico</b>	<b>Idios</b>
<input type="checkbox"/> Tabla ANOVA	<input type="checkbox"/> Gráfico de Medias	<b>Almacén</b>
<input type="checkbox"/> Tabla de Medias	<input type="checkbox"/> Gráfico de Caja y Bigotes	<b>Ayuda</b>
<input checked="" type="checkbox"/> Pruebas de Múltiple Rangos	<input type="checkbox"/> Gráficos de Residuos	
<input type="checkbox"/> Verificación de Varianza	<input type="checkbox"/> Gráfico Análisis de Medias (ANOM)	

**Opciones de Ventana...**

**Opciones de Análisis...**

Deshacer	Ctrl+Z
Cortar	Ctrl+X
Copiar	Ctrl+C
Copiar como Metafile	
Copiar con Vínculo	
Pegar	Ctrl+V
Imprimir...	F4
Vista Preliminar...	Mayú+F3
Copiar Ventana a StatReporter	
Copiar Análisis a StatReporter	
Mostrar XML...	

**ANOVA Simple - Tiempo por TipoDiscos**

**Pruebas de Múltiple Rangos para Tiempo por TipoDiscos**

Método: 95,0 porcentaje LSD

TipoDiscos	Casos	Media	Grupos Homogéneos
SAS	5	109,0	X
SATA	5	114,8	X
IDE	5	136,0	X

Contraste	Sig.	Diferencia	+/- Límites
IDE - SAS	*	27,0	11,1723
IDE - SATA	*	21,2	11,1723
SAS - SATA		-5,8	11,1723

\* indica una diferencia significativa.

**Opciones Prueba de Rangos Múltiples**

Método:

☒ **LSD**

☐ Tukey HSD

☐ Scheffe

☐ Bonferroni

☐ Student-Newman-Keuls

☐ Duncan

Nivel de Confianza:

95,0 %

**Aceptar**

**Cancelar**

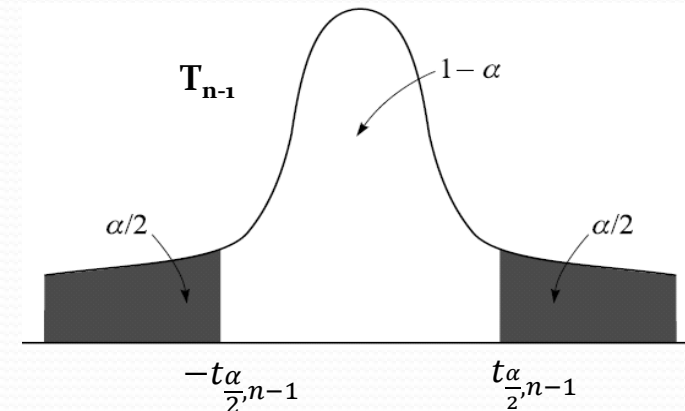
**Ayuda**

# Resumen: Test t y Test ANOVA

## Test T

- $H_0$ : Rendimiento A  $\equiv$  Rendimiento B ( $d_i \sim \mathcal{N}(\bar{d}_{real}, \sigma^2)$ ,  $\bar{d}_{real} = 0$ ).
- $t_{exp} = \frac{\bar{d} - \bar{d}_{real}}{s/\sqrt{n}} \sim T_{n-1}$  siendo  $\bar{d} = \frac{\sum_{i=1}^n d_i}{n}$   $s = \sqrt{\frac{\sum_{i=1}^n (d_i - \bar{d})^2}{n-1}}$
- Valor-p  $\approx$  Prob ( $H_0$  podría ser cierta).
- Rechazamos  $H_0$  para un nivel de confianza  $(1 - \alpha) \cdot 100(\%)$  si:
  - ✓ valor-p  $< \alpha$
  - ✓  $t_{exp} \notin [-t_{\frac{\alpha}{2}, n-1}, t_{\frac{\alpha}{2}, n-1}]$
  - ✓  $0 \notin [\bar{d} - \frac{s}{\sqrt{n}} \times t_{\frac{\alpha}{2}, n-1}, \bar{d} + \frac{s}{\sqrt{n}} \times t_{\frac{\alpha}{2}, n-1}]$

Exp.	tA	tB	$d_i = tA_i - tB_i$
P <sub>1</sub>	tA <sub>1</sub>	tB <sub>1</sub>	d <sub>1</sub>
P <sub>2</sub>	tA <sub>2</sub>	tB <sub>2</sub>	d <sub>2</sub>
...	...	...	...
P <sub>n</sub>	tA <sub>n</sub>	tB <sub>n</sub>	d <sub>n</sub>



## Test ANOVA

- $H_0$ : Rendimiento de todos los niveles del factor es equivalente ( $\varepsilon_j = 0$ ,  $j = 1, \dots, n_{niv}$ )  $\rightarrow$  El factor no influye en el rendimiento.
- $F_{exp} \equiv \frac{SSA/(n_{niv}-1)}{SSE/(n_{niv} \times (n_{rep}-1))} \sim F_{n_{niv}-1, n_{niv} \times (n_{rep}-1)}$
- Valor-p  $\approx$  Prob ( $H_0$  podría ser cierta).
- Rechazamos  $H_0$  para un nivel de confianza  $(1 - \alpha) \cdot 100(\%)$  si valor-p  $< \alpha$ .