

HaplotypeCaller GVCF Workflow



Dr. Matthieu J. Miossec (@RealMattJM)

Bioinformatics analyst @ Wellcome Centre for human genetics



UNIVERSITY OF
OXFORD

Programa Unidad 9

- 17 de mayo (lunes) – Introducción a la genómica en la nube
- 19 de mayo (miércoles) – Introducción a la plataforma Terra
- 24 de mayo (lunes) – Otras herramientas en Terra (GATK mejores practicas)



- El miércoles, en un par de horas, logramos:
 - subir archivos BAM/BAI.
 - organizar nuestros datos de alineamiento y de referencia.
 - Parametrizar y ejecutar HaplotypeCaller, extrayendo variantes.
- Ya podríamos descargar el resultado de este trabajo (.VCF) y seguir trabajando localmente o seguir trabajando en Terra usando el Jupyter Notebook o RStudio (más tarde)

Descubrimiento de variantes



- GATK → Genome Analysis ToolKit
(<https://gatk.broadinstitute.org/>)



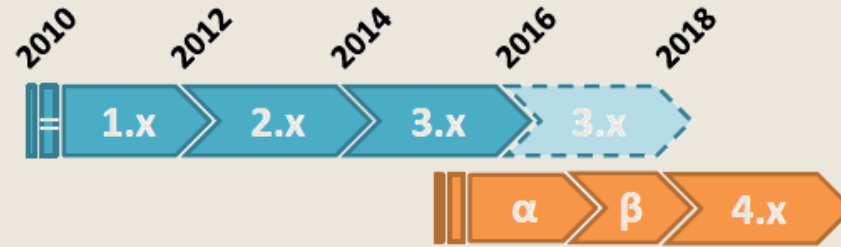
- Suele se usa el término para hablar estrictamente del llamador de variante → **HaplotypeCaller**
- Pero en realidad GATK corresponde a mucho más, especialmente en su ultima versión GATK 4...

GATK 4

- Disponible desde 2018.

- Última versión: 4.2.0.0

- Open source: <https://github.com/broadinstitute/gatk/>



- Rediseñado para mejor **velocidad**, **escabilidad** y **versatilidad**.
- Incorpora herramientas para llamar varios tipos de variantes:
 - En varios estados de finalización

| | GERMLINE | SOMATIC |
|---------------------|----------------------|----------------------------|
| SNPs & INDELS | HaplotypeCaller GVCF | MuTect2 |
| Copy Number | gCNV (beta) | CNV (ModelSegments) (beta) |
| Structure Variation | SVDDiscovery (beta) | (planned) |

GATK 4 en la nube con



- GATK 4 también fue especialmente rediseñado para funcionar en la nube.
 - GATK 4 interface directamente con el API de Google
 - ➔ Puede leer y acceder a partes de un archivo directamente del espacio en que esta almacenado sin copiarlo entero.
 - Todos los ‘workflows’ de GATK, siguiendo los ‘Best Practices’ están disponible en forma ejecutable en Terra.

Suggested Tools

haplotypcaller-gvcf-gatk4

Runs HaplotypeCaller from GATK4 in GVCF mode on a single sample

joint-discovery-gatk4

Implements the joint discovery and VQSR filtering

processing-for-variant-discovery-gatk4

Find Additional Tools



Dockstore

Browse WDL workflows in Dockstore, an open platform used by the GA4GH for sharing Docker-based tools

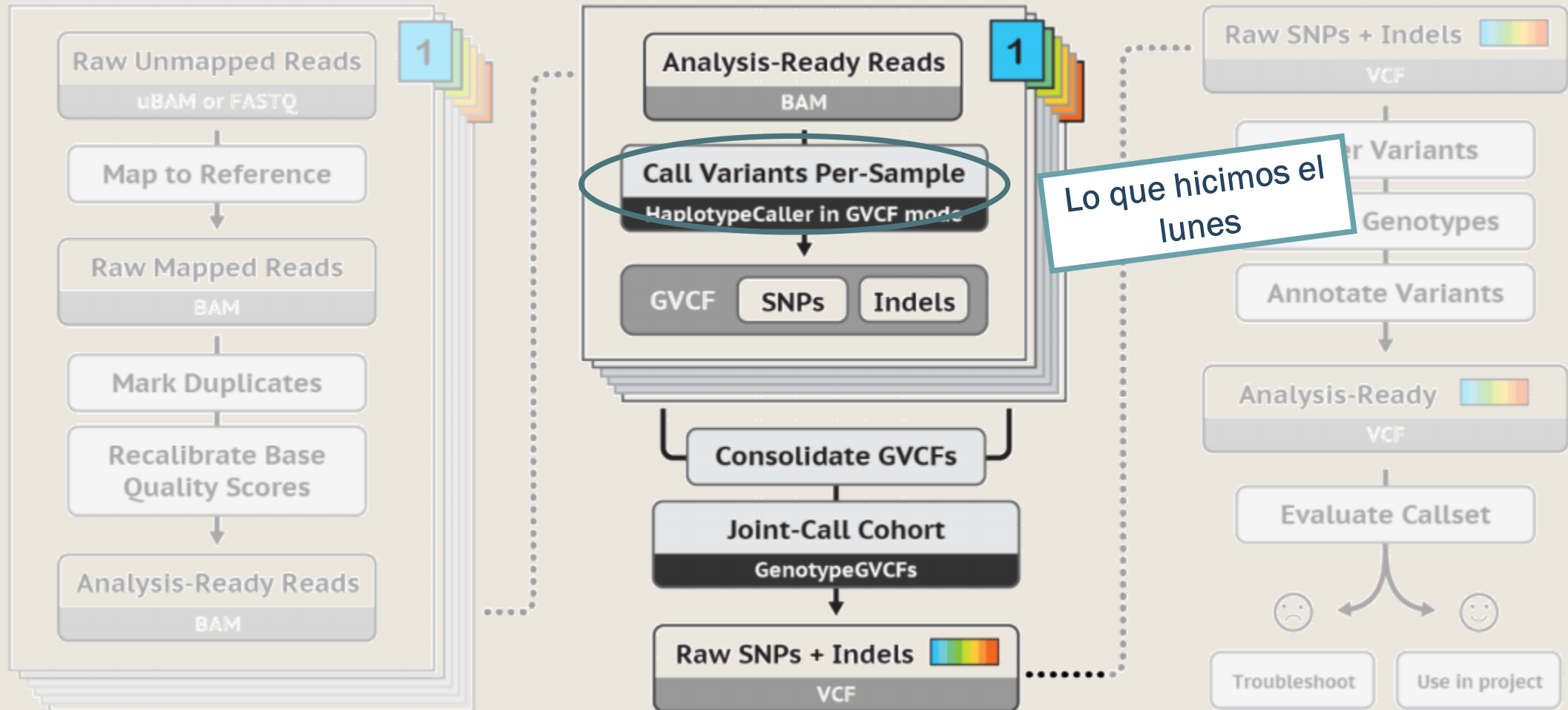
¿Por qué GATK HaplotypeCaller GVCF?

■ GATK HaplotypeCaller

- Considerado el **criterio de excelencia** en termino de llamado de variantes.
- Respaldado por un **gran equipo de soporte**.
- **Documentación** muy detallada.
- **Reproducible** gracias a los 'Best Practices'.
 - Especialmente reproducibles si están ejecutado en Terra!
- Diseñado para ser muy escalable → va a permitir llamar variantes en cohortes enormes (ej. +800 pacientes [exomas] con enfermedad congénita cardiaca)

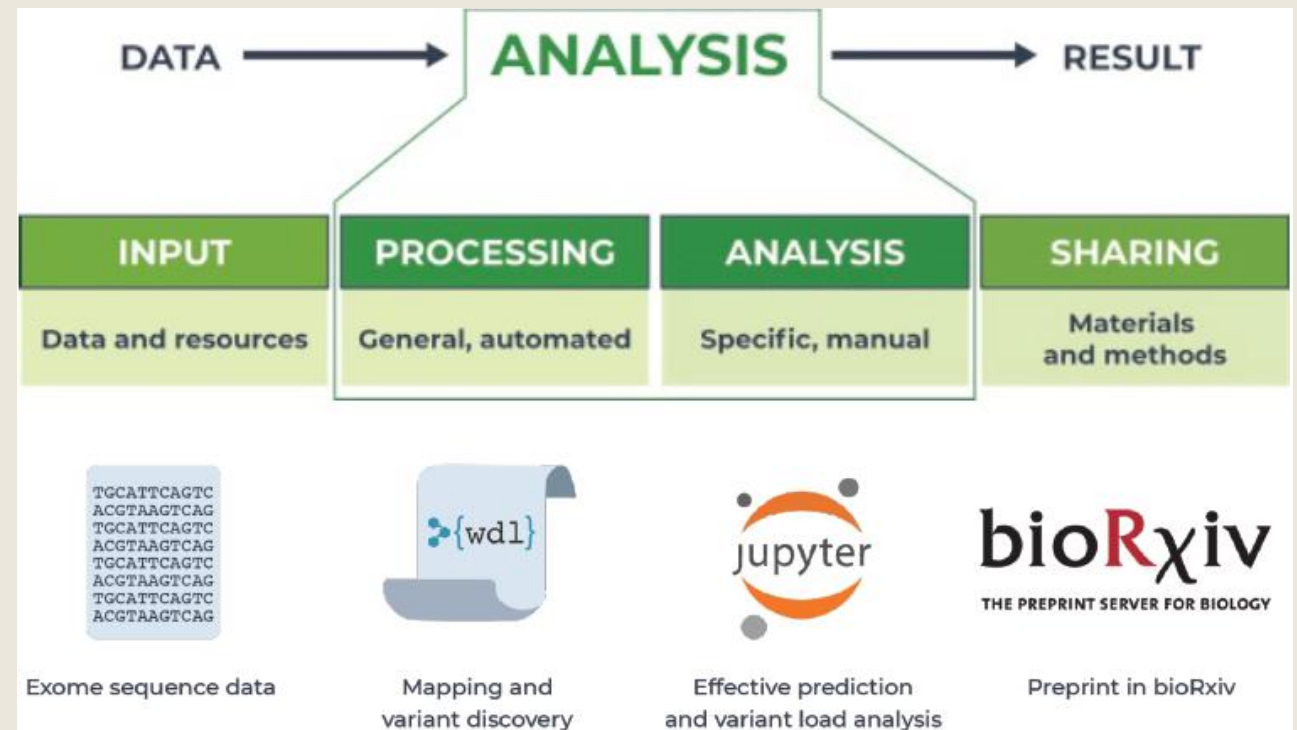
GATK HaplotypeCaller GVCF

‘Best Practices’



Análisis en Tiempo Real

- En cada análisis llega un momento cuando el investigador quiere consultar los datos generados, aplicar filtros, reorganizar con tests estadísticos....



- Y así regresamos a trabajar con el **Jupyter notebook!**



- El Jupyter notebook es un documento interactivo que puede contener y ejecutar líneas de código.
 - Originalmente pensado para tres lenguaje (**J**ulia, **P**ython y **R**), pero ahora cubre mucho más.
- En Terra, notebooks están disponible para Python 2, 3 y R.
 - Perfecto para descifrar y visualizar nuestros datos una vez la etapas de computo intensivo terminadas.

Configuración en Terra

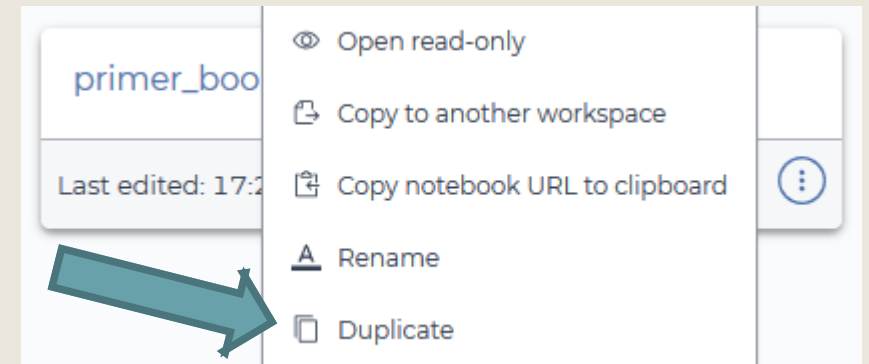
- Para funcionar adentro de Terra, el notebook necesita un maquina virtual (abr. ing. VM).
 - Puede ser consultado sin VM, pero como documento fijo.
 - Cada vez que usamos el notebook configuramos una VM según nuestras necesidades computacionales.
 - Espacio disco
 - CPU
 - Memoria
 - Con cada ejecución tenemos una estimación del costo por hora.
 - Se apaga después de una media hora sin actividad pero mejor parrar la ejecución nosotros.



Cuidado: El notebook en un 'workspace' compartido



- Por razones de seguridad, la VM que uno genera para usar un notebook es solo accesible por la persona que lo genera, no puede ser compartida.
- **Consecuencia** = Si dos investigadores en un espacio de trabajo compartido están trabajando sobre el mismo notebook, uno puede borrar el trabajo del otro a través del autoguardado.
 - Por eso que existe la posibilidad de duplicar notebook!



Extra: Acceso al 'Workspace' en Jupyter notebook con *os.environ*.

- Para consultar los datos de nuestro espacio de trabajo más directamente necesitamos hacer establecer el siguiente.

Python

```
import os

BILLING_PROJECT_ID = os.environ['WORKSPACE_NAMESPACE']
WORKSPACE = os.environ['WORKSPACE_NAME']
bucket = os.environ['WORKSPACE_BUCKET']
```

R

```
project <- Sys.getenv('WORKSPACE_NAMESPACE')
workspace <- Sys.getenv('WORKSPACE_NAME')
bucket <- Sys.getenv('WORKSPACE_BUCKET')
```

Extra: Guardar datos del Jupyter notebook en el Bucket

- Durante la ejecución del notebook, podemos subir y guardar archivos usando las clases que existen en Python y R para estas operaciones, pero...
 - Al cerrar la VM, estos archivos dejan de existir!
- Para guardarlos en el cubo Google:

Python

```
#Copy the file data.txt in the notebook into the bucket
!gsutil cp ./data.txt $bucket

#Run list command to see if file is in the bucket
!gsutil ls $bucket
```

R

```
#Copy the file data.txt in the notebook into the bucket
system(paste0("gsutil cp ./data.txt ",bucket),intern=TRUE)

#Run list command to see if file is in the bucket
system(paste0("gsutil ls ",bucket),intern=TRUE)
```