Supplementary Material and Methods

SweGen: A whole-genome data resource of genetic variability in a cross-section of the Swedish population

Adam Ameur, Johan Dahlberg, Pall Olason, Francesco Vezzi, Robert Karlsson, Marcel Martin, Johan Viklund, Andreas Kusalananda Kähäri, Pär Lundin, Huiwen Che, Jessada Thutkawkorapin, Samuel Lampa, Mats Dahlberg, Jonas Hagberg, Niclas Jareborg, Inger Jonasson, Åsa Johansson, Lars Feuk, Joakim Lundeberg, Ann-Christine Syvänen, Sverker Lundin, Daniel Nilsson, Björn Nystedt, Patrik Magnusson, Ulf Gyllensten

Table of contents

Supplementary Tables 2
Supplementary Figures 3
1. Single nucleotide variant (SNV) and indel analysis 5
  1.1 Commands for SNV and indel detection 5
  1.2 Commands for joint variant calling 7
2. Structural variation (SV) analysis 10
  2.1 Commands for structural variation detection 10
  2.2 Commands for structural variation filtering 10

Supplementary Tables

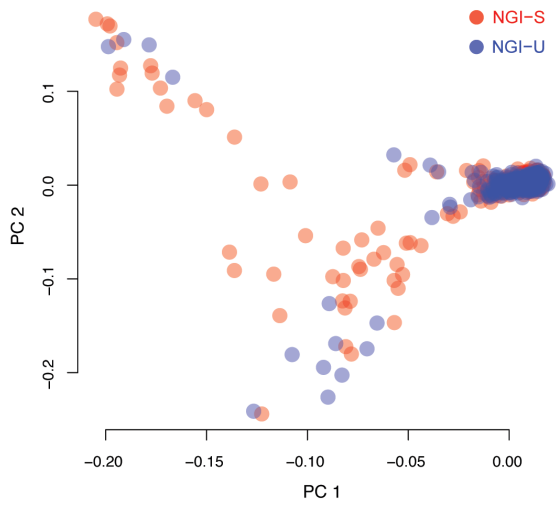
Supplementary Table S1. Regions known to harbor strong long-range LD.

chromosome	LD region start (hg19)	LD region end (hg19)
5	44 Mb	51.5 Mb
6	25 Mb	33.5 Mb
8	8 Mb	12 Mb
11	45 Mb	57 Mb

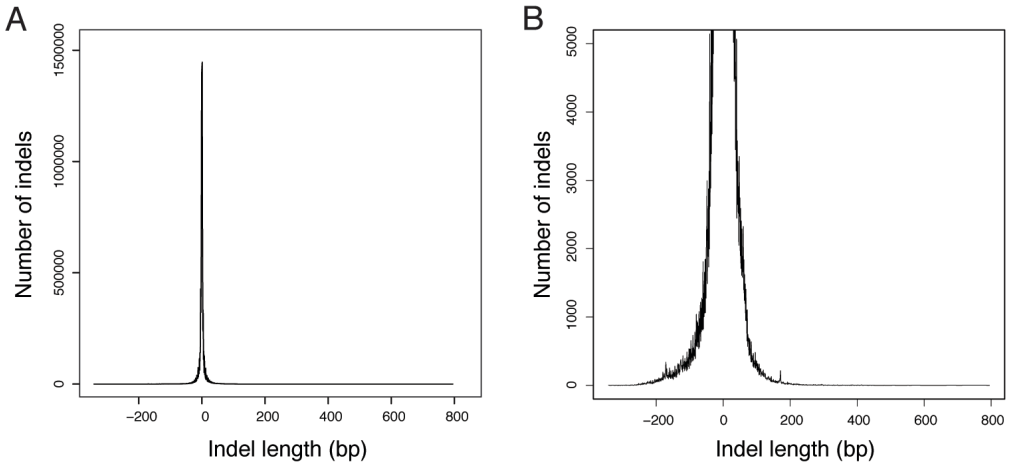
Supplementary Table S2. List of exons with zero coverage for at least 50% of their bases, in at least 5 of 8 samples in the pilot run.

chromosome	exon start (hg19)	exon end (hg19)	refseq	gene
1	110230418	110230531	NM_000561	GSTM1
1	110230792	110230867	NM_000561	GSTM1
1	110231295	110231359	NM_000561	GSTM1
1	110231670	110231751	NM_000561	GSTM1
1	110231847	110231947	NM_000561	GSTM1
1	110232893	110232988	NM_000561	GSTM1
1	110233076	110233186	NM_000561	GSTM1
1	152573138	152573562	NM_178434	LCE3C
1	152586287	152586574	NM_178433	LCE3B
10	51970583	51970664	NM_001143974	ASAH2
2	97854824	97854852	NM_001164315	ANKRD36
2	97854955	97855027	NM_001164315	ANKRD36
2	98152763	98152835	NM_025190	ANKRD36B
2	98152931	98152959	NM_025190	ANKRD36B
2	98156497	98156569	NM_025190	ANKRD36B
2	98156664	98156692	NM_025190	ANKRD36B
2	98162296	98162324	NM_025190	ANKRD36B
20	1577986	1578928	NM_001135844	SIRPB1
20	1579468	1579582	NM_001135844	SIRPB1
20	1584456	1584788	NM_001135844	SIRPB1
20	1585388	1585705	NM_001135844	SIRPB1
6	32485154	32485529	NM_002125	HLA-DRB5
6	32485831	32485854	NM_002125	HLA-DRB5
6	32486333	32486443	NM_002125	HLA-DRB5
7	74379083	74379195	NM_001145063	GATSL1
8	7415850	7415958	NM_001136572	FAM90A7P
8	7806651	7806777	NM_001193630	ZNF705B

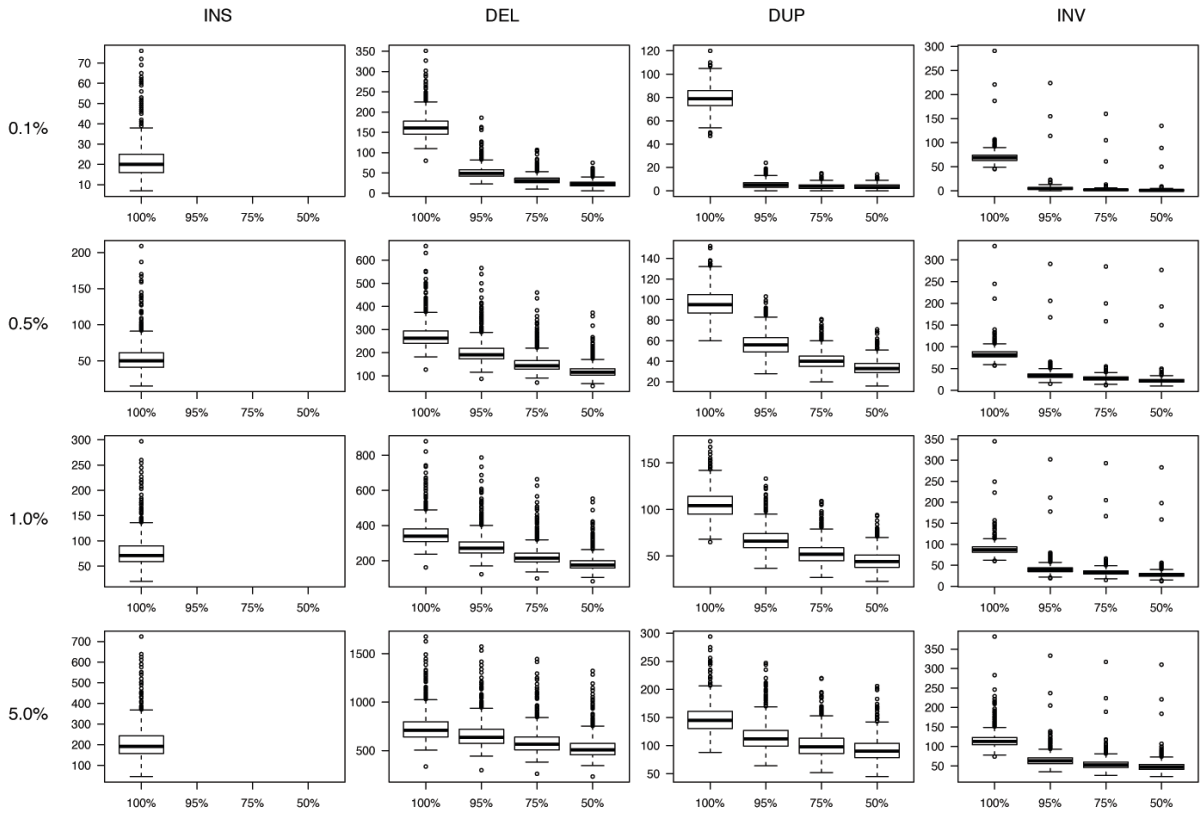
Supplementary Figures



Supplementary Figure S1. PCA of variation between sequencing sites in the 1000 WGS samples. 509 of the SweGen samples were sequenced at NGI-S (red) and 491 samples at NGI-U (blue). The PCA plot shows the distribution of genetic variation based on ~650k common SNPs from the Illumina OmniExpress chip. There are no clear biases detected in the first two principal components when comparing the two sites.



Supplementary Figure S2. Length distribution for indel variants in the SweGen dataset. A) The x-axis displays the length of indels (negative for insertions, positive for deletions) and the total number of variants in the SweGen dataset are shown on the y-axis **B)** Same data as in panel A, but with the y-axis truncated at 5000 to obtain higher resolution of the longer indels.



Supplementary Figure S3. Filtering of structural variation in the SweGen dataset. Each row shows the number of structural variants remaining in a WGS sample after filtering all events occurring filtering at a specific frequency in the SweGen dataset (0.1%, 0.5%, 1% and 5%). For each of the 1000 genomes, INS, DEL, DUP and DEL calls were filtered against the SweGen SV frequencies to produce a box plot distribution for the number of SVs remaining after filtering. For each of the SV types, four different analyses were performed requiring a reciprocal overlap of 100%, 95%, 75% and 50% between SVs in order to be filtered. Since partial overlaps are not defined for INS (see Methods), only the 100% data is shown for these events.

1. Single nucleotide variant (SNV) and indel analysis

1.1 Commands for SNV and indel detection

Below are the exact commands that were executed by the SNV analysis pipeline. To increase readability paths, execution environment specific parameters (e.g. JVM temporary paths) and other very long strings such as read group information have been removed. All parameters of importance to the output results have been kept. The analysis was coordinated using Piper (https://github.com/NationalGenomicsInfrastructure/piper)

 # Multiple instances of bwa mem has been run, one for each flowcell and lane on which the sample was run

set -o pipefail; bwa mem -M -t 16 -R "<read group info>" human_g1k_v37.fasta 2004002226_S2_L004_R2_001.fastq.gz 2004002226_S2_L004_R1_001.fastq.gz | samtools view -Su - | samtools sort -@ 16 -m 2G - 01_raw_alignments/2004002226.BHNGLYCCXX.2004002226.4;samtools index 01_raw_alignments/2004002226.BHNGLYCCXX.2004002226.4.bam

qualimap --java-mem-size=112G bamqc -bam 01_raw_alignments/2004002226.BHNGLYCCXX.2004002226.5.bam --paint-chromosome-limits --genome-gc-distr HUMAN -outdir 02_preliminary_alignment_qc/2004002226.BHNGLYCCXX.2004002226.5.qc/ -nt 16 --skip-duplicated --skip-dup-mode 0

 # The group of bam files related to a sample were merged in this step, with one "INPUT" per bam file.

java -cp <java_class_path> picard.sam.MergeSamFiles INPUT=<input bam 1> INPUT=<input bam 2> [...] OUTPUT=04_merged_aligments/2004002226.bam COMPRESSION_LEVEL=5 VALIDATION_STRINGENCY=SILENT SO=coordinate CREATE_INDEX=true USE_THREADING=true

java -cp <java_class_path> org.broadinstitute.gatk.engine.CommandLineGATK -T RealignerTargetCreator -I 04_merged_aligments/2004002226.bam -R human_g1k_v37.fasta -nt 16 -o 05_processed_alignments/2004002226.intervals -known dbsnp_138.b37.vcf -known Mills_and_1000G_gold_standard.indels.b37.vcf -known 1000G_phase1.indels.b37.vcf -mismatch 0.0

java -cp <java_class_path> org.broadinstitute.gatk.engine.CommandLineGATK -T IndelRealigner -I 04_merged_aligments/2004002226.bam -R human_g1k_v37.fasta -compress 5 -known dbsnp_138.b37.vcf -known Mills_and_1000G_gold_standard.indels.b37.vcf -known 1000G_phase1.indels.b37.vcf -targetIntervals 05_processed_alignments/2004002226.intervals -o 05_processed_alignments/2004002226.clean.bam -model USE_READS

java -cp <java_class_path> picard.sam.MarkDuplicates INPUT=05_processed_alignments/2004002226.clean.bam OUTPUT=05_processed_alignments/2004002226.clean.dedup.bam COMPRESSION_LEVEL=5 VALIDATION_STRINGENCY=SILENT CREATE_INDEX=true M=05_processed_alignments/2004002226.metrics

java -cp <java_class_path> org.broadinstitute.gatk.engine.CommandLineGATK -T BaseRecalibrator -I 05_processed_alignments/2004002226.clean.dedup.bam -R human_g1k_v37.fasta -nct 16 -knownSites dbsnp_138.b37.vcf -o 05_processed_alignments/2004002226.pre_recal.table -cov ReadGroupCovariate -cov QualityScoreCovariate -cov CycleCovariate -cov ContextCovariate

java -cp <java_class_path> org.broadinstitute.gatk.engine.CommandLineGATK -T PrintReads -I 05_processed_alignments/2004002226.clean.dedup.bam -R human_g1k_v37.fasta -baq CALCULATE_AS_NECESSARY -BQSR 05_processed_alignments/2004002226.pre_recal.table -compress 5 -nct 16 -o 05_processed_alignments/2004002226.clean.dedup.recal.bam

java -cp <java_class_path> org.broadinstitute.gatk.engine.CommandLineGATK -T BaseRecalibrator -I 05_processed_alignments/2004002226.clean.dedup.recal.bam -R human_g1k_v37.fasta -nct 16 -knownSites dbsnp_138.b37.vcf -o 05_processed_alignments/2004002226.post_recal.table -cov ReadGroupCovariate -cov QualityScoreCovariate -cov CycleCovariate -cov ContextCovariate

java -cp <java_class_path> org.broadinstitute.gatk.engine.CommandLineGATK -T HaplotypeCaller -I 05_processed_alignments/2004002226.clean.dedup.recal.bam -R human_g1k_v37.fasta -dcov 250 -nct 16 -variant_index_type LINEAR -variant_index_parameter 128000 -o 07_variant_calls/2004002226.clean.dedup.recal.bam.genomic.vcf.gz -D dbsnp_138.b37.vcf -ERC GVCF -stand_call_conf 30.0 -stand_emit_conf 10.0 -pairHMM LOGLESS_CACHING -pcrModel CONSERVATIVE

java -cp <java_class_path> org.broadinstitute.gatk.engine.CommandLineGATK -T GenotypeGVCFs -R human_g1k_v37.fasta -nt 16 -V

07_variant_calls/2004002226.clean.dedup.recal.bam.genomic.vcf.gz -o

07_variant_calls/2004002226.clean.dedup.recal.bam.raw.vcf.gz -D dbsnp_138.b37.vcf

1.2 Commands for joint variant calling

Joint variant calling has been performed following guidelines at:
- https://www.broadinstitute.org/gatk/guide/article?id=3893
- http://gatkforums.broadinstitute.org/gatk/discussion/2805/howto-recalibrate-variant-quality-scores-run-vqsr

Joint calling is a resource intense analysis, both in terms of storage and computation. In order to reduce the running time and as suggested by GATK guidelines the following strategy has been followed:
- Set-up of the dataset:
    - Divide the 1000 samples in 5 batches of 200 samples each
    - Divide the human genome in 52 non-overlapping intervals of 60Mbp
- Step1: CombineGVCF:
    - For each interval and for each batch run CombineGVFC step
- Step2: GenotypeGVCFs
    - For each interval run GenotypeGVCFs on all the batches
- Step3: CatVariants
    - Merge the non overlapping intervals
- Step4: VQSR
- Run VQSR as explained in http://gatkforums.broadinstitute.org/gatk/discussion/2805/howto-recalibrate-variant-quality-scores-run-vqsr

The complete list of commands is available at this link

https://github.com/SciLifeLab/joint_variant_calling/tree/master/SweGen

Examples of commands run:

Step 1: CombineGVCF
This command is run once for every batch i and every interval j

```
java -cp <java_class_path>  org.broadinstitute.gatk.engine.CommandLineGATK  -T CombineGVCFs bam -R human_g1k_v37.fasta \
-V sample_1_batch_i.vcf.gz \
-V sample_2_batch_i.vcf.gz \
…
-V sample_N_batch_i.vcf.gz \
-L interval_j.txt \
-o SweGen_batch_i_interval_j.g.vcf.gz
```

Step2: GenotypeGVCFs
This commands is run once for every interval and takes as input the N batches from step1

```
java -cp <java_class_path>  org.broadinstitute.gatk.engine.CommandLineGATK \
-T GenotypeGVCFs bam -R human_g1k_v37.fasta \
```

- nt 16 \
--max_alternate_alleles 6 \
-D dbsnp 138.b37.vcf \
-V SweGen_batch_1_interval_j.g.vcf.gz \
-V SweGen_batch_2_interval_j.g.vcf.gz \
…
-V SweGen_batch_N_interval_j.g.vcf.gz \
-L interval_j.txt \
-o SweGen_joincalled_interval_j.g.vcf.gz \

Step3: CatVariants
This command is executed only once and merges the K vcf files from step2

java -cp <java_class_path> org.broadinstitute.gatk.engine.CommandLineGATK \
-T GenotypeGVCFs bam -R human_g1k_v37.fasta \
-assumeSorted \
-V SweGen_joincalled_interval_1.g.vcf.gz \
-V SweGen_joincalled_interval_2.g.vcf.gz \
...
-V SweGen_joincalled_interval_K.g.vcf.gz \
-out SweGen_joincalled.g.vcf.gz

Step4: VQSR
This step follow best practice described at
http://gatkforums.broadinstitute.org/gatk/discussion/2805/howto-recalibrate-variant-quality-scores-run-vqsr

java -cp <java_class_path> org.broadinstitute.gatk.engine.CommandLineGATK \
-T VariantRecalibrator \
-R human_g1k_v37.fasta \
-nt 16 \
-resource:hapmap,known=false,training=true,truth=true,prior=15.0 hapmap_3.3.b37.vcf \
-resource:omni,known=false,training=true,truth=true,prior=12.0 1000G_omni2.5.b37.vcf \
-resource:1000G,known=false,training=true,truth=false,prior=10.0
1000G_phase1.snps.high_confidence.b37.vcf \
-resource:dbsnp,known=true,training=false,truth=false,prior=2.0 dbsnp_138.b37.vcf \
-an QD \
-an MQ \
-an MQRankSum \
-an ReadPosRankSum \
-an FS \
-an SOR \
-an DP \
-an InbreedingCoeff \
-mode SNP \
-tranche 100.0 -tranche 99.9 -tranche 99.0 -tranche 90.0 \
-input SweGen_joincalled.g.vcf.gz \
-recalFile SweGen_joincalled.snp.recal \
-tranchesFile SweGen_joincalled.snp.tranches

java -cp <java_class_path> org.broadinstitute.gatk.engine.CommandLineGATK \
-T ApplyRecalibration \

```
-R human_g1k_v37.fasta \
-nt 16 \
-input SweGen_joincalled.g.vcf.gz \
-mode SNP \
--ts_filter_level 99.0 \
-recalFile SweGen_joincalled.snp.recal \
-tranchesFile SweGen_joincalled.snp.tranches \
-o SweGen_joincalled.recal_snp_raw_indels.vcf.gz



#### Prepare recalibration parameters for Indels
java -cp <java_class_path>  org.broadinstitute.gatk.engine.CommandLineGATK \
-T VariantRecalibrator  \
-R human_g1k_v37.fasta \
-nt 16 \
--maxGaussians 4 \
-resource:mills,known=false,training=true,truth=true,prior=12.0
Mills_and_1000G_gold_standard.indels.b37.vcf \
-resource:dbsnp,known=true,training=false,truth=false,prior=2.0 dbsnp_138.b37.vcf \
-an QD \
-an DP \
-an FS \
-an SOR \
-an ReadPosRankSum \
-an MQRankSum \
-an InbreedingCoeff \
-mode INDEL \
-tranche 100.0 -tranche 99.9 -tranche 99.0 -tranche 90.0 \
-input SweGen_joincalled.recal_snp_raw_indels.vcf.gz \
-recalFile SweGen_joincalled.indel.recal \
-tranchesFile SweGen_joincalled.indel.tranches


java -cp <java_class_path>  org.broadinstitute.gatk.engine.CommandLineGATK \
-T ApplyRecalibration  \
-R human_g1k_v37.fasta \
-nt 16 \
-input SweGen_joincalled.recal_snp_raw_indels.vcf.gz \
-mode INDEL \
--ts_filter_level 99.0 \
-recalFile SweGen_joincalled.indel.recal \
-tranchesFile SweGen_joincalled.indel.tranches \
-o SweGen_joincalled.recal_snp_recal_indels.vcf.gz
```

2. Structural variation (SV) analysis

2.1 Commands for structural variation detection

configManta.py --normalBam <SAMPLE>.bam --referenceFasta human_g1k_v37.fasta --runDir <SAMPLE>.mantaDir
cd <SAMPLE>.mantaDir && ./runWorkflow.py -m local -j 16

2.2 Commands for structural variation filtering

BEDTools 2.23.0. swegen-beds/ contains Manta VCFs converted to BED.

(for f in swegen-beds/*.bed.gz; do zcat $f | uniq ; done) | sort -k1,1V -k2,2n -k3,3n | uniq -c | awk -vOFS='\t' '{print $2, $3, $4, $5, $1}' | gzip > structural-variants.bed.gz

Then choose a TYPE (DEL, DUP, INS or INV); the BEDPATH to the sample BED to be filtered; the reciprocal OVERLAP (1, 0.95, 0.75, 0.5) and at which FREQUENCY threshold a SV must occur to be used for filtering. Create the filtered BED with:

bedtools intersect -wa -v -f $OVERLAP -r -a <(zcat $BEDPATH|grep $TYPE) -b <(zcat structural-variants.bed.gz|awk "\$5>=$FREQUENCY"| grep $TYPE)