Supporting Online Material for:

Genome sequencing of *C. elegans* balancer strains reveals previously unappreciated complex genomic rearrangements

Tatiana Maroilley[1,2], Stephane Flibotte[3], Francesca Jean[1,2], Victoria Rodrigues Alves Barbosa[1,2], Andrew Galbraith[1,2], Afiya Razia Chida[1,2], Filip Cotra[1,2], Xiao Li[1,2], Larisa Oncea[1,2], Mark Edgley[4], Don Moerman[4], Maja Tarailo-Graovac[1,2]

Corresponding author:

Dr. Maja Tarailo-Graovac – maja.tarailograovac@ucalgary.ca

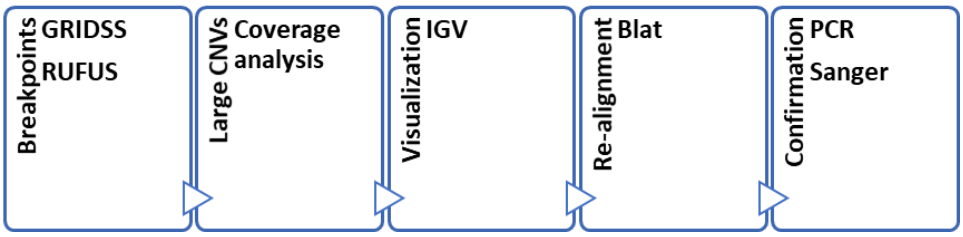
This PDF file includes: Supplemental Methods and Supplemental Figures S1-S10.

Contents

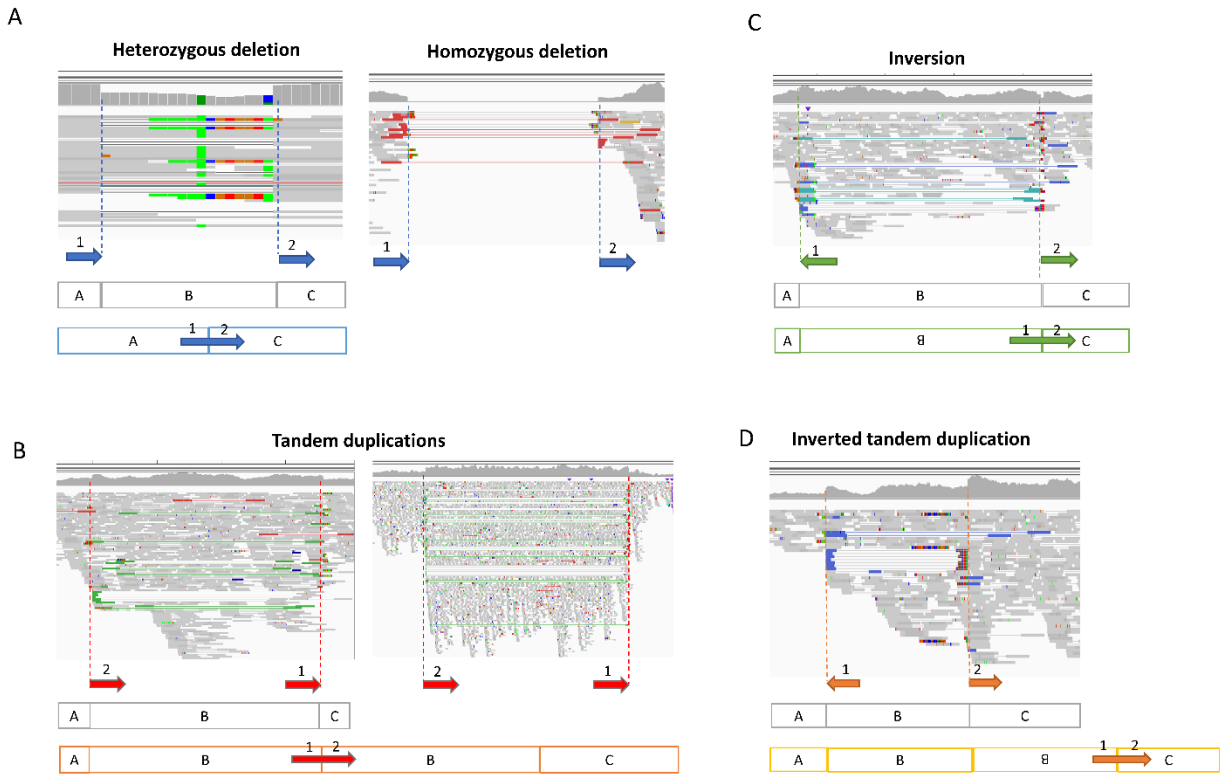
Supplemental Tables

- Supplemental_Table_S1: List of the strains used in this manuscript and previous studies published by our group.
- Supplemental_Table_S2: Data from CGC.
- Supplemental_Table_S3: Sequencing information for the strains sequenced in the project.
- Supplemental_Table_S4: Description of the breakpoints found in balancers, with primers and Sanger sequences.
- Supplemental_Table_S5: Description of the non-balancer events detected in each genome.
- Supplemental_Table_S6: List of genes overlapped by each balancer.
- Supplemental_Table_S7: List of variants detected in CB3475.
- Supplemental_Table_S8: List of variants detected in KR1876.

Workflow



Supplemental Figure S1: Bioinformatics and experimental workflow from Maroilley et al., 2021



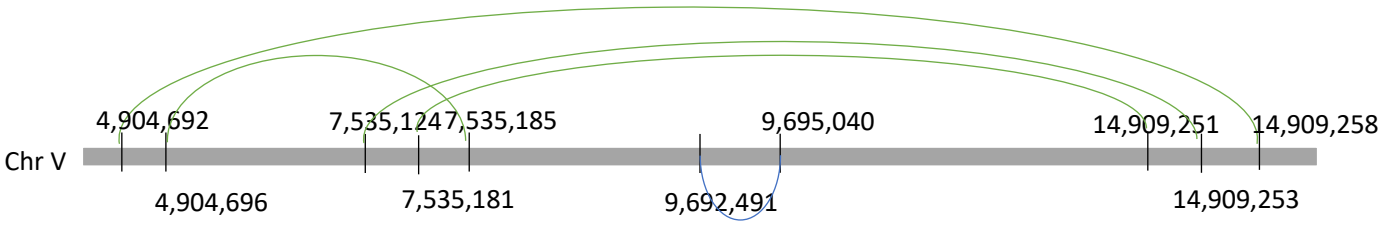
Supplemental Figure S2: Signature of split reads visualized by IGV and reflecting the structure of a rearrangement. A) In blue, representation of a split read suggesting a deletion. The second part of the read gets aligned with the same orientation at the end of the deleted region. B) In red, representation of a split read suggesting a tandem duplication. Both parts of the split reads align in the same orientation. The second part of the read aligns at the end of the duplicated region, while the first part aligns before, at the start of the duplicated region. C-D) In green and orange, representation of split reads suggesting either inversion or inverted tandem duplication. Only a large change or no change in copy number can differentiate the events. In both cases, both parts of the split read align in opposite orientation. As for direct duplication, the second part of the read aligns at the end of the duplicated region, while the first part aligns before, at the start of the duplicated region. Heterozygous deletion: I:14,338,214-14,338,233 (20 bp in length; detected in RM2431); Homozygous deletion: IV:5,370,371-5,371,720 (1,349 bp in length; detected in RW6002); Homozygous tandem duplication: I:10,565,986-10,577,022 (11,042 bp in length; detected in CZ1072); Heterozygous tandem duplication: I:1,307,000-1,308,796 (1,796 bp in length; detected in RM2431); Inversion: IV:4,399,382-4,401,161 (1,783 bp in length; detected in BC1217); Inverted tandem duplication: V:16,305-17,193 (888 bp in length; detected in SP423).

As shown in recent papers for four *C. elegans* balancers (Maroilley et al. 2021; Edgley et al. 2021;

Flibotte et al. 2021), srWGS contains enough information to explore complex rearrangements. Here, we

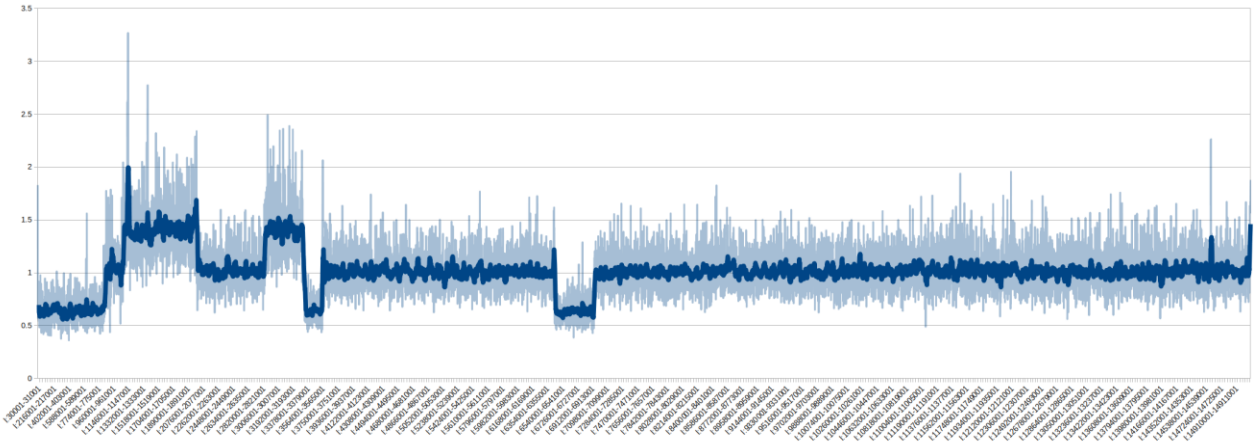
used the workflow as described in Maroilley et al. (2021) (Supplemental Figure S1). In brief, we combined callings from GRIDSS (Cameron et al. 2017) and RUFUS (Ostrander et al. 2018). We filtered out breakpoints that are also present in N2 strain by visual assessment on Integrative Genomic Viewer (IGV) (Robinson et al. 2011). In addition, we filtered by visual inspection false positive breakpoints that do not show read signatures of a true breakpoint (Supplemental Figure 2). Then, we re-aligned reads around the breakpoints to list split reads supporting the breakpoint (Supplemental Figure 2). The mapping and orientation of each part of split reads is informative of the type of structural variant (Supplemental Figure 2). We then designed primers based on split reads and run PCR and Sanger sequencing to confirm the breakpoints. In case of suspected copy number variants (CNVs) not supported by split reads (free duplication, large deletion or copy number gain/loss due to complex rearrangement), we performed a coverage analysis by plotting read coverage by window of 1kb which allowed the definition of approximate boundaries for such events.

mnC1 (strain SP127)

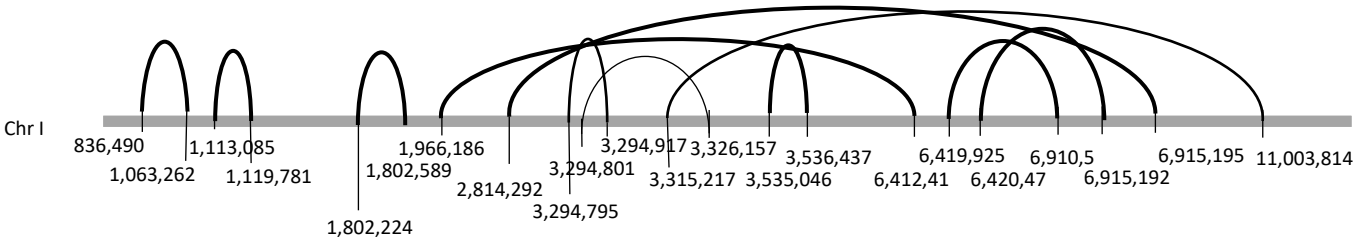


Supplemental Figure S3: Linear representation of the breakpoints identified in mnC1

hDf8 (strain KR1233)

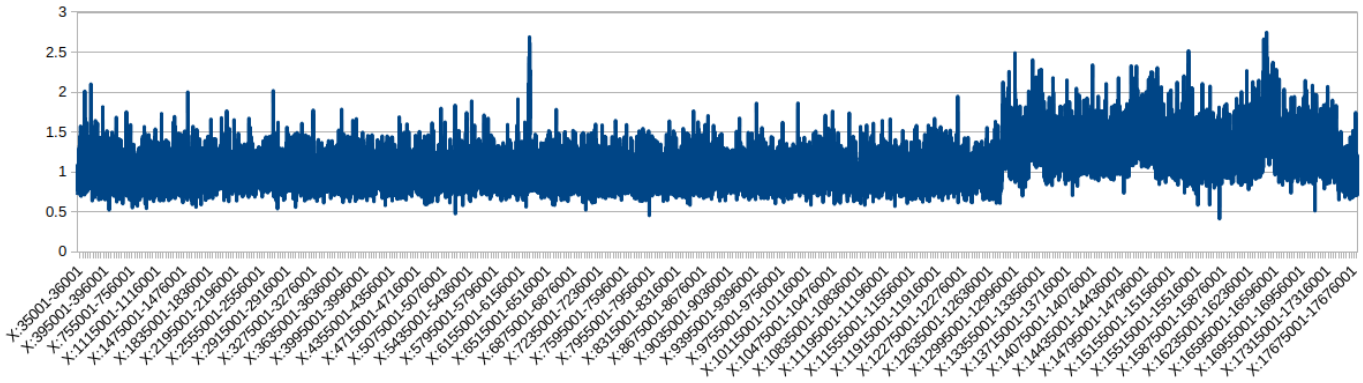


Supplemental Figure S4: Variation in coverage along Chr I in the strain KR1233, carrying the balancer hDf8. The read coverage was calculated with SAMtools depth by window of 1 kb and normalized (light blue). The line in dark blue is a trend line representing a moving average of 20 points.

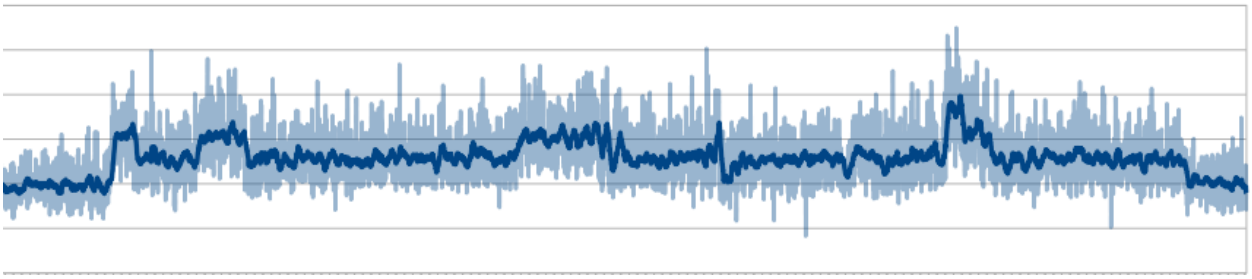


Supplemental Figure S5: Linear representation of the breakpoints identified in hDf8. Bold junctions have been confirmed by PCR and Sanger sequencing.

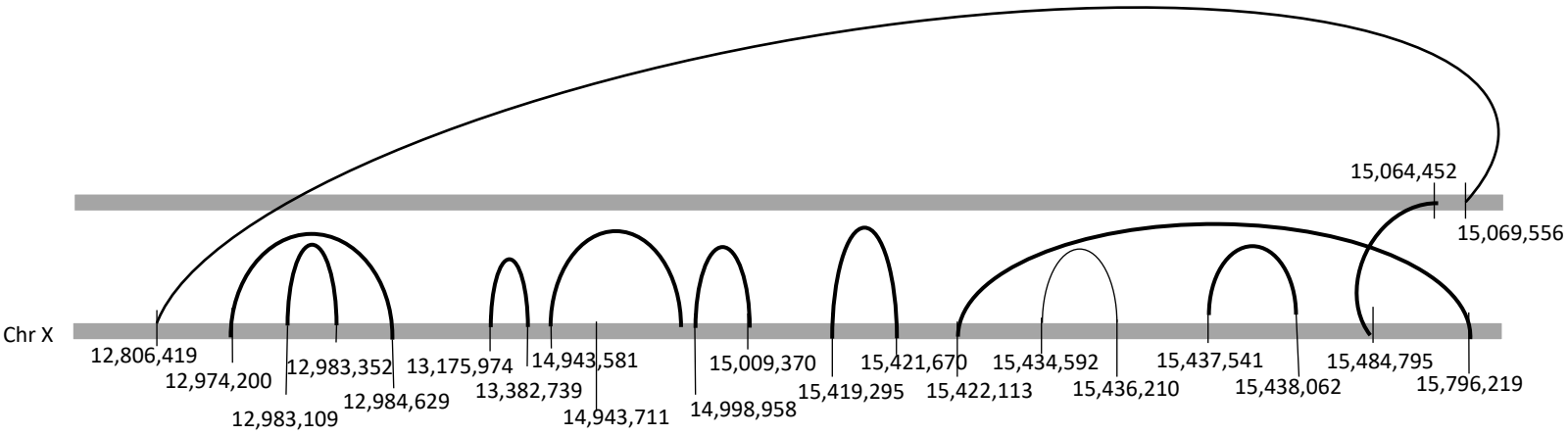
mnDp3 (strain SP123)



Supplemental Figure S6: Variation in coverage along Chr X in the strain SP123, carrying the balancer mnDp3. The coverage increased to a 1.2 ratio on the right end of Chr X showing the free duplication. The read coverage was calculated with SAMtools depth by window of 1 kb and normalized.



Supplemental Figure S7: Zoom in on the free duplication region, showing variations in the coverage suggesting complex rearrangement of the free duplicate of Chr X. The read coverage was calculated with SAMtools depth by window of 1 kb and normalized (light blue). The line in dark blue is a trend line representing a moving average of 20 points.



Supplemental Figure S8: Linear representation of the breakpoints identified in mnDp3. Bold junctions have been confirmed by PCR and Sanger sequencing.

Experimental validation

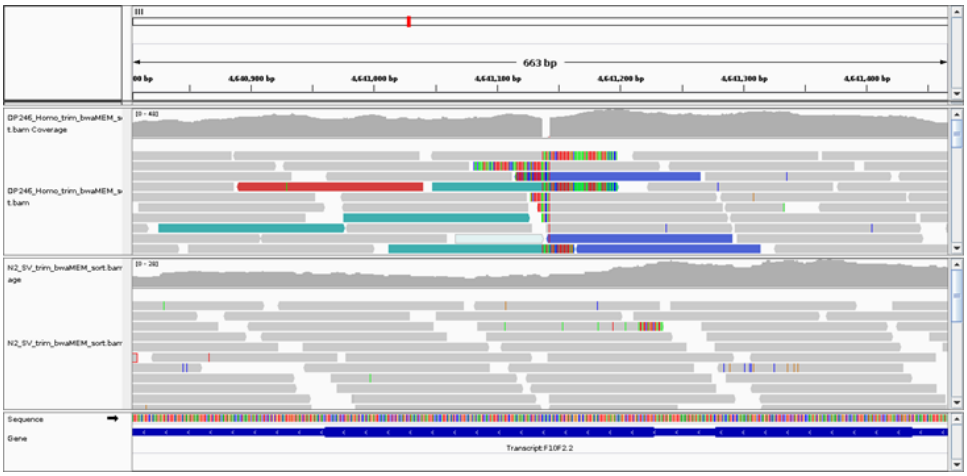
Most breakpoints were confirmed by PCR and the junctions at the breakpoints were reported with Sanger Sequencing. When available, PCR gels are in Supplemental Material > PCR gels section. Sanger sequences are available in Supplemental Table S4 for balancer breakpoints and in Supplemental Table S5 for additional events.

We could not design primers for some balancers breakpoints. In the case of *sDf22* and *hDf15*, both deletions, no split reads were detected at the breakpoints. In a similar way, no split reads are present in case of free duplication. In addition, a translocate duplication only shows split reads related to the translocation, not the duplication. But, in all three cases, the coverage analysis helped find an approximate position for the boundaries of each copy number variation.

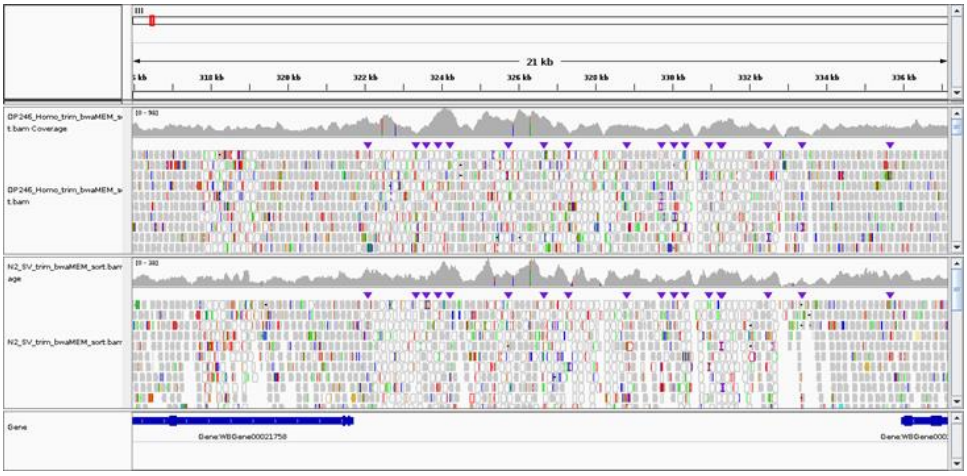
Still, in the case of *sDf22*, we designed primers surrounding the region with appearing loss of copies and obtained a band at the expected size. Sequencing by Sanger the PCR product confirmed the left breakpoint of the deletion, but the rest of the Sanger sequence of bad quality (Ns), limiting the validation of the right breakpoints.

For *mT1(II;III)*, reciprocal translocation between Chr II and Chr III, the breakpoint on Chr II falls into a repetitive region, which made the experimental validation difficult. The primers designed based on those breakpoints (Supplemental Tables S4 and S5) led to the amplification of a band at the expected size, but the sequence obtained by Sanger contained mismatches that troubled the confirmation of the exact breakpoint on Chr II. The breakpoint on Chr III is, however, confirmed.

For *sC1*, large inversion on Chr III, the right breakpoint falls into a highly repetitive region with multiple alignment reads covering the region (Figure S10). Despite our best efforts, we could not get a primer design to confirm the breakpoints.

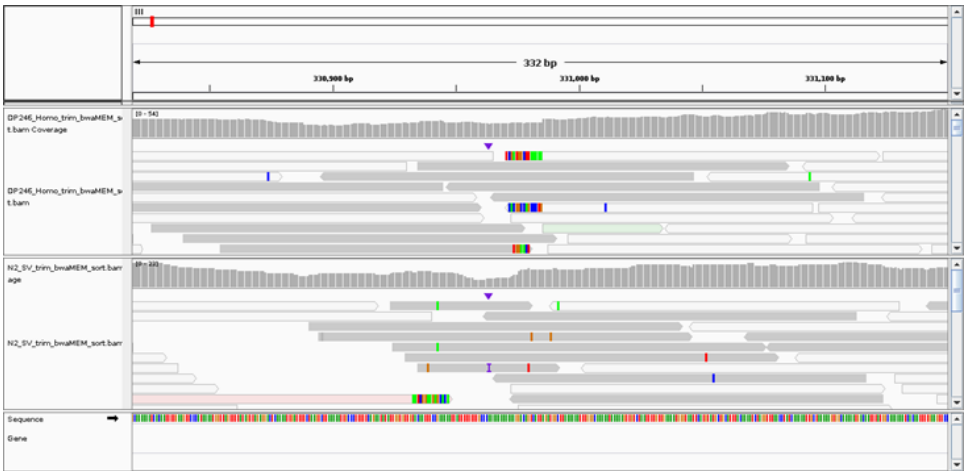


A

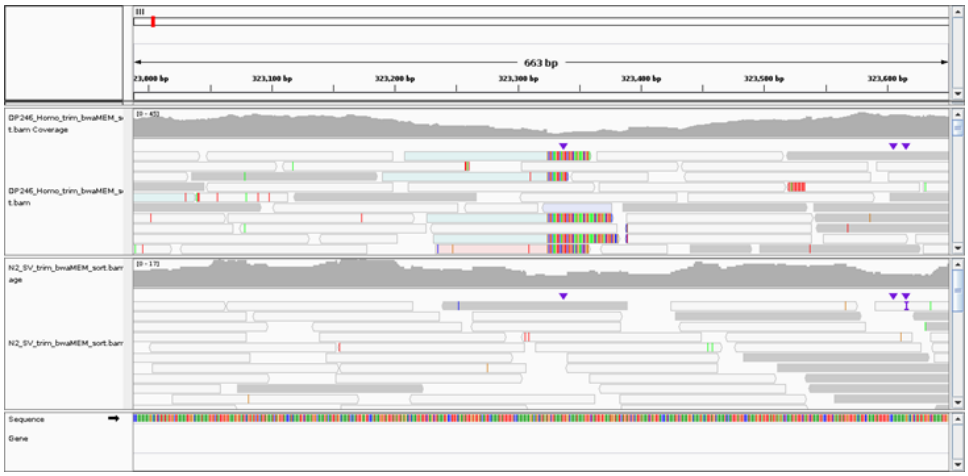


B

C



D



Supplemental Figure S9: IGV screenshots of sC1 breakpoints and region surrounding sC1 breakpoints. A) Clear signature of reads for the right breakpoint of the inversion (III:4641137). B) Region surrounding left breakpoint. C-D) Two possible genomic locations based on bioinformatics analyses for sC1 left breakpoint (III:323321 or III:330985)

For balancers based on complex genomic rearrangements with breakpoints close to one another, designing primers was challenging (*stDp2, e917*). It is also possible that our analyses missed a few breakpoints, explaining why our designs failed at amplifying the breakpoints junctions. In addition, read coverage based on srWGS often showed clear gain or loss of copies between breakpoints in CGRs, but with no supporting split reads. In those cases, it is possible that those gains/losses of copies are

secondary events, consequences of larger breakages and rearrangements, explaining the absence of split reads.

Due to the extensive list of SVs and CGRs that were uncovered in the various genomes of this study, we decided to confirm by PCR and Sanger Sequencing only a subset of them. The list of experimentally confirmed events in Table 3. The complete list of additional events detected by RUFUS and manually curated in silico is available in Supplemental Table S5.

Accuracy of srWGS

srWGS, compared to PCR, allows the exploration of the entire genome, far beyond the balanced region, at a reduced cost and without much more time invested in the analysis. In addition, it is unbiased as it does not rely on probes' design. Finally, it is equally accurate in the detection of small and large rearrangements, if the proper downstream analyses are applied, while techniques like FISH and karyotype would only reveal large events.

Over all the breakpoints reported in this study, we evaluate the accuracy of srWGS at mapping precisely the breakpoints by comparing the genomic positions reported with genomics to the genomic positions revealed by Sanger sequencing. The genomic mapping with srWGS was perfectly accurate for 104 breakpoints, validated by Sanger and part of additional structural variants and complex genome rearrangements we described in this study. However, for 14 other breakpoints, the genomic position we identified from srWGS was not completely accurate and was mapped away from the position reported by Sanger on average by 4 bp [1-26 bp]. This is partially imputable to the manual interpretation step of the re-alignment of the reads, sometime inaccurate due to the rearrangements at the breakpoint junction.

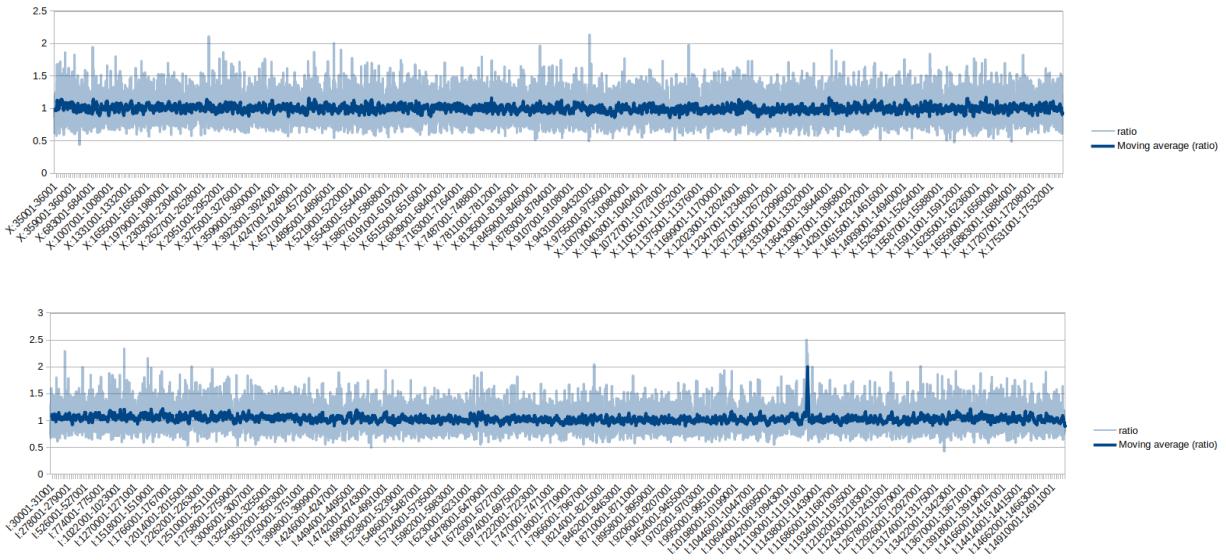
Balancer Derivatives or Limitation of srWGS

Two additional balancers were initially included in this study: *mnDp1* and *hT3.*

The balancer *mnDp1* was described as a translocated duplication of Chr X to Chr V, with a severe reduction of recombination frequency from *unc-60* (V:1475086-1479576) to *dpy-11* (V:6511583-6515240). To retrieve the breakpoints for this balancer, we sequenced the genome of the SP423 [*mnDp1 (X;V)/+ V; unc-3(e151) let-2(mn153) X*] strain. Interestingly in this case, our analysis did not corroborate most of those findings. The coverage analysis based on short read WGS we performed confirmed the duplication of Chr X from around X:13,794,000 to the right end of Chr X (ratio = 1.5). In addition, we observed a stretch of heterozygous SNPs from X:13884091 to the right end of Chr X (35/36 SNPs are heterozygous). But, despite our best efforts, we could not retrieve signs of a translocation of the duplicate to Chr V. In addition, out of the 43 SNPs and indels we have detected on Chr V, 15 were heterozygous, but they were not consecutive on the chromosome. The absence of a clear loss of homozygosity rules out the fact the Chr V is balanced in the SP423 strain we sequenced. It is possible that in this genome, the translocation broke, because of the instability of such rearrangement and/or srWGS failed at detecting breakpoints of a rearrangement in this genome. However, we identified two additional rearrangements – one at the left end of Chr V and the second on Chr X around 5.6 Mb. Finally, a breakpoint close to Chr X: 13,785,731 is highly suggestive of an insertion, but with very few supporting reads.

The balancer *hT3* has been described as a very stable translocation effectively balancing for left portion of Chr I from left end to around *let-363*, and the right portion of Chr X from the right end to between *dpy-7* and *unc-3*. We selected the strain KR1876 *[+/hT3[dpy-5(e61)] I; dpy-7(e88) unc-3(e151)/hT3 X]* to study this balancer. We have sequenced this strain twice, in two different facilities, once with heterozygous genomes and once with homozygous genomes. In either case, we did not uncover

breakpoints suggestive of a translocation (I;X). No variation in coverage was observed (Supplemental Figure S10).



Supplemental Figure S10: Line plots representing read coverage normalized along Chr I and Chr X for KR1876

However, the presence of heterozygous variants on the right end of Chr I and the left end of X corroborates previous characterization of the balancer (see Supplemental Table S8). It is then possible that the breakpoints are in repetitive regions, making their detection by srWGS difficult.

Re-sequencing *qC1* and *hT2* balancers

The molecular characterization of the balancers *qC1* and *hT2* have been recently published (Edgley et al., 2021 and Flibotte et al., 2021). In this study, we have sequenced strains also described carrying those two balancers (GE2722 *[cyk-1(t1568) unc-32(e189)/qC1 [dpy-19(e1259) glp-1(q339)] III; him-3(e1147) IV.]* and VC571 *[+/hT2 I; kin-18(ok395)/hT2 [bli-4(e937)] III.]*), but different from the ones used in Edgley at al., 2021 and Flibotte et al., 2021. For *qC1*, we were able to retrieve most of the breakpoints

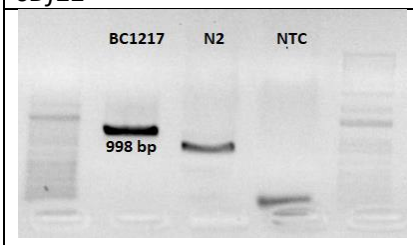
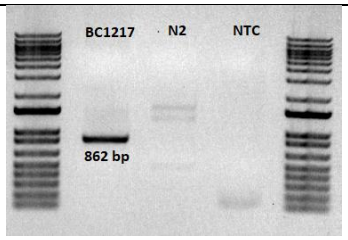
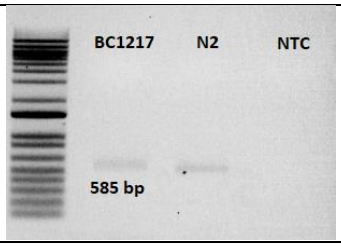
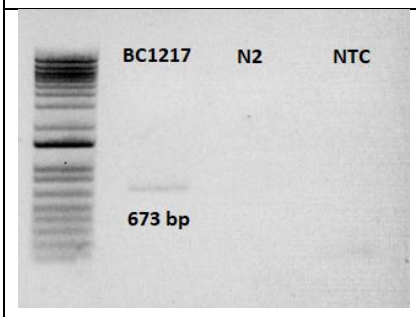
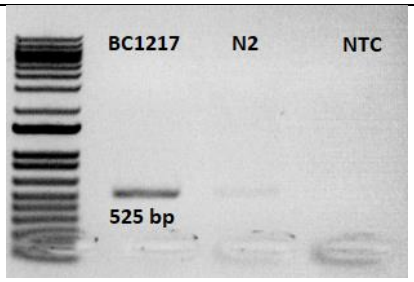
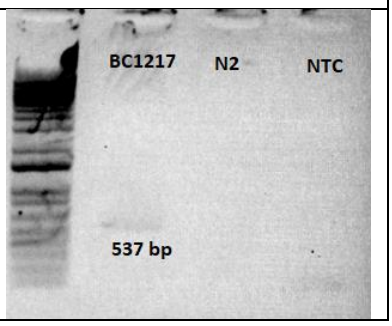
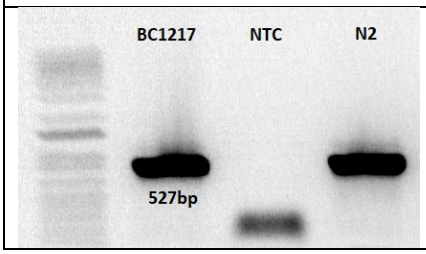
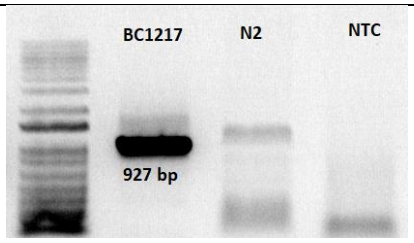
described and we confirmed them with PCR and Sanger sequencing. We also found additional rearrangements in the GE2722, with some of them overlapping the balanced regions by *qC1*. For *hT2,* we sequenced the strain VC471 in which no breakpoints and junctions were revealing a reciprocal translocation (I;III) suggesting that for this strain, the translocation might have broken.

Independent analyses of *nT1* and *mIn1*

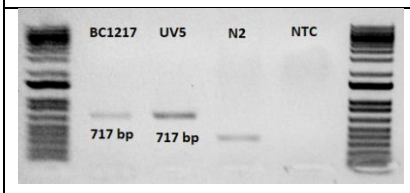
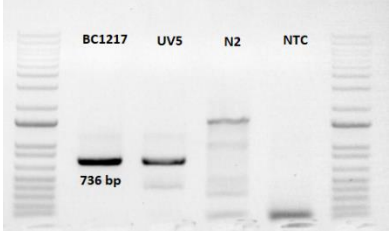
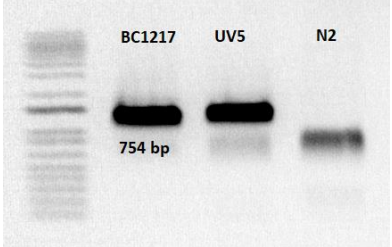
Both *nT1* and *mIn1* were explored independently in other strains than the ones included in this manuscript (respectively BC1217 and VC1771). Both analyses showed the same characterization and breakpoint mapping. We have only reported here results and experimental confirmation in the strains BC1217 and VC1771. In addition, the UV5 strain (carrying *nT1*) has been used in PCR gels as a control for *nT1* but it was not sequenced.

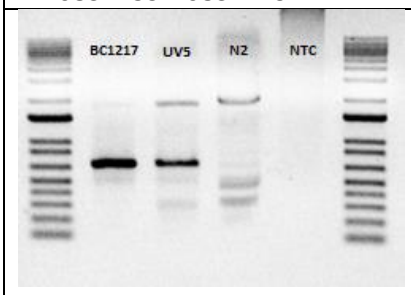
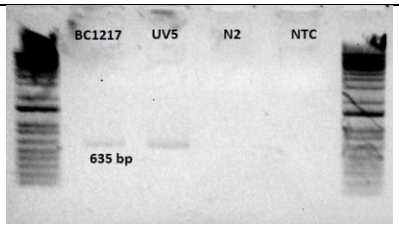
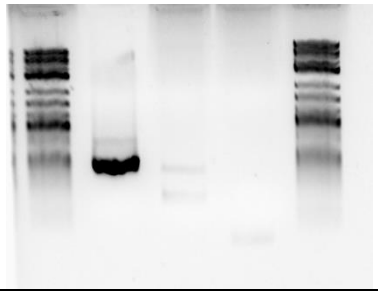
PCR gels pictures

BC1217

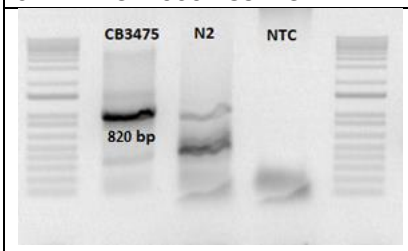
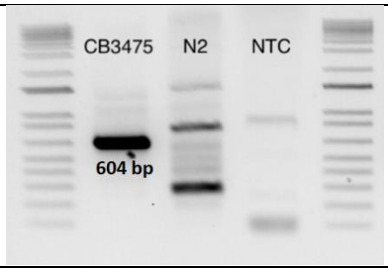
sDf22	Deletion IV:12418048-12417282	Inversion IV:4399382-4401161
		
Inversion: IV:8065618-8066587	Homozygous Tandem Duplication III:5059444-5063035	Tandem Duplication IV:14576313-14582569
		
Tandem Duplication X:1509128-1515794	Homozygous Tandem Duplication X:14993793-14974538	
		

BC1217

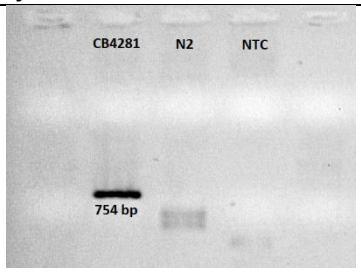
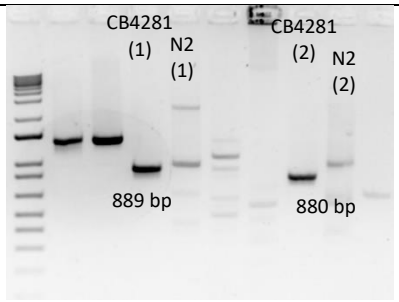
nT1 IV:1901208-V:11072802 or V:11081514	*nT1* IV:11255068 -V:13866656 and IV:11254917 -V:5515463	*nT1* IV:1901518-V:11077404
		

nT1 V:11078600-16831547 and V:16831759-16832779	*nT1* V:5622382-13861523	*nT1* V:1108443-16821528 (band 813 bp)
		

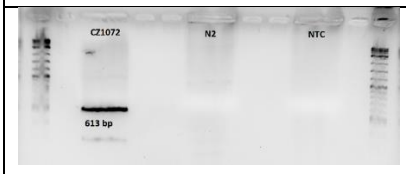
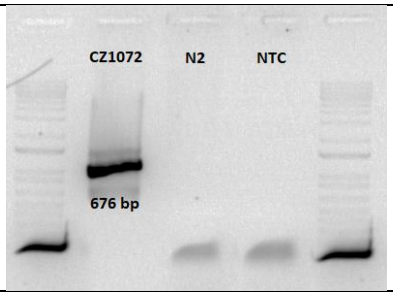
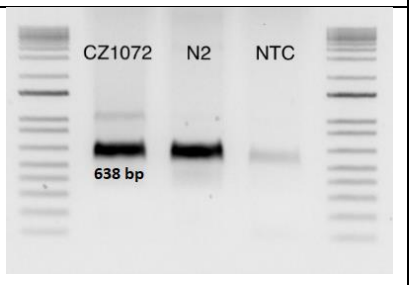
CB3475

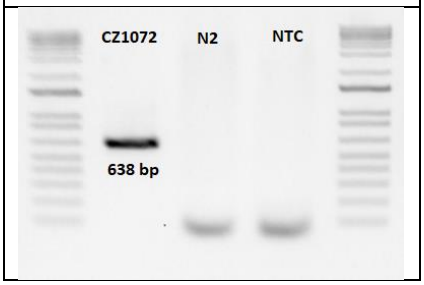
szT1 X:2314606-2597434	*szT1* I:7631470-X:2314605
	

CB4281

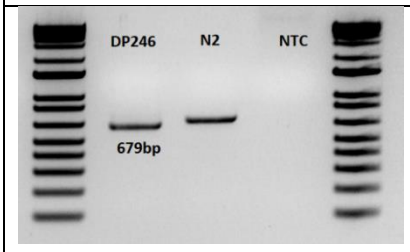
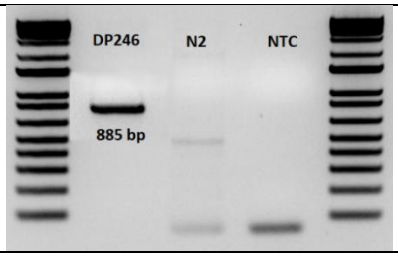
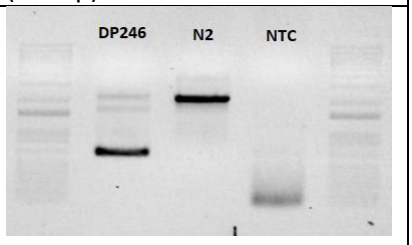
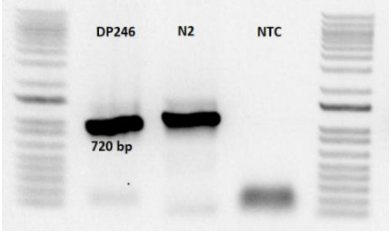
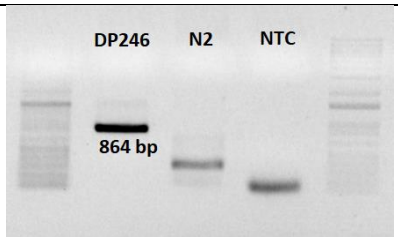
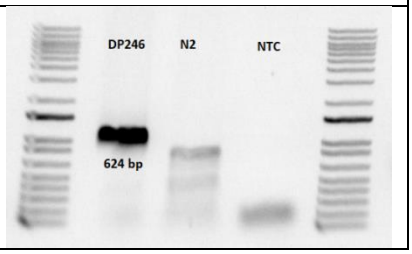
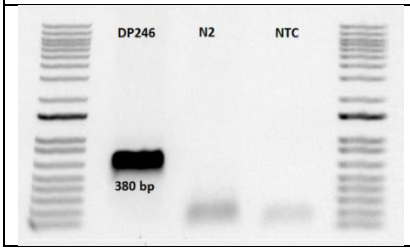
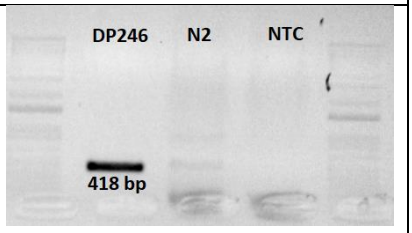
eDf43 V:3466648-3469548	*eDf43* (1) V:3466642-3483655 and (2) V:3466718-3714670
	

CZ1072

e917 Inversion: V: 4489199-14577009	Inverted tandem duplication I: 15271369-15273155	Tandem Duplication III: 3891219-3898488
		

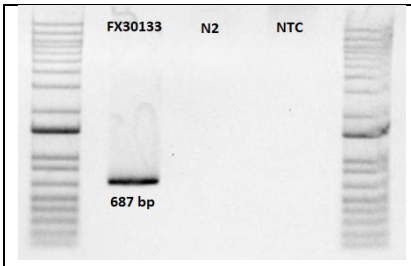
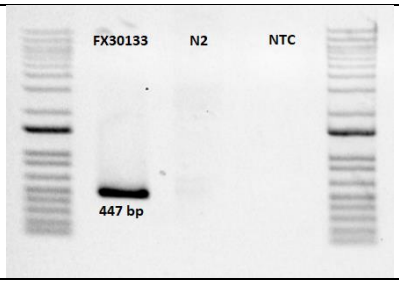
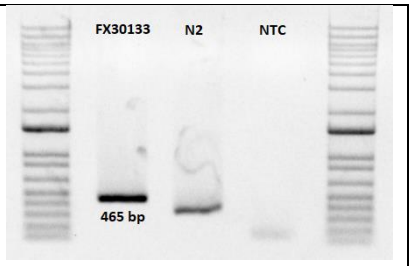
Tandem Duplication I:10565986-10577022


DP246

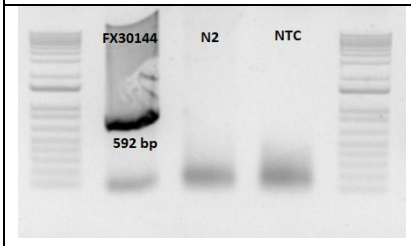
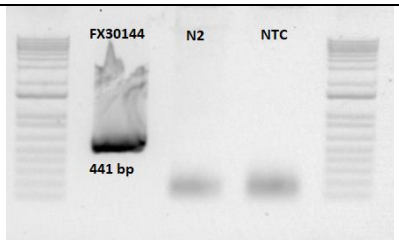
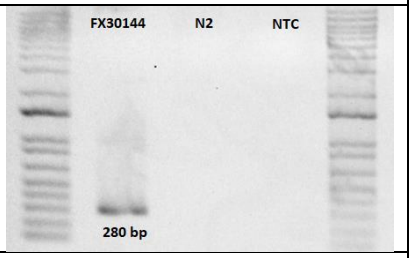
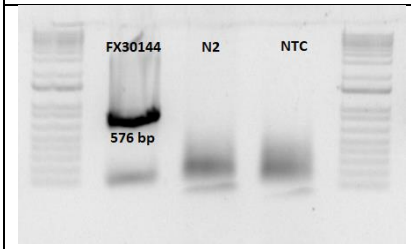
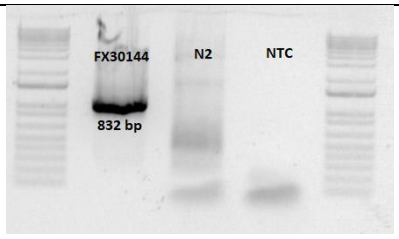
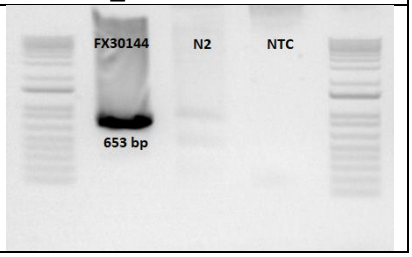
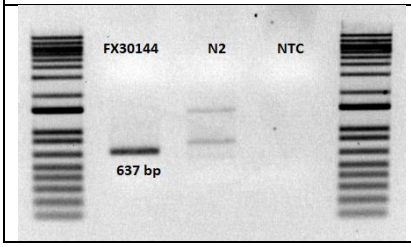
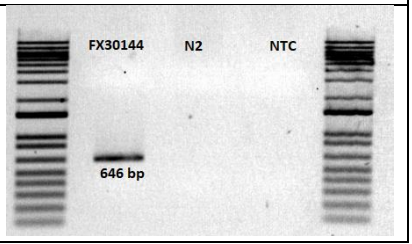
Deletion III:5402384-540245	Tandem duplication V:16914510-16928379	Deletion III:11120257-11121729 (601bp)
		
Deletion IV: 3052310-3052413	Tandem duplication III:11112143-11129064	Deletion I: 12824847-12825125
		
Inverted tandem duplication X:1708726-4450957	Inverted tandem duplication X:1845385-4409188	
		

FX30133

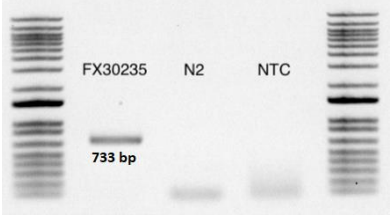
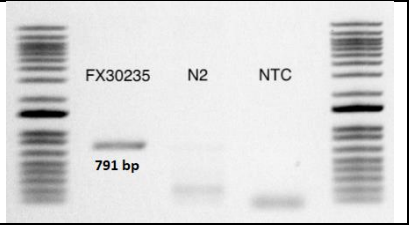
Deletion V:15826754-15886806	*tmC3* Inversion V:6484110-12142846	*tmC3* Inversion V: 8938061-12197769

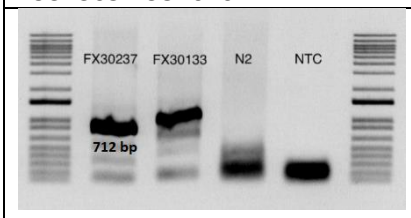
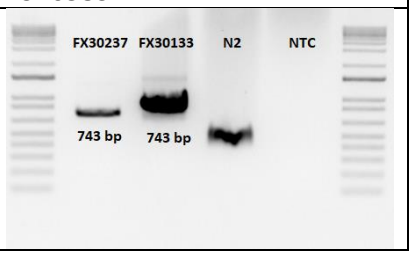
FX30144

Duplication II: 5131837-5805620	Deletion II:5136381-5806749	CGR IV:16464403-16465989
		
tmC6 Inversion II:2331869-2758154	*tmC6* Inversion II:2332242-2758243	*tmC6* Inversion II:2671237-6912298_
		
CGR IV:16469112-16510345	CGR IV:16462855-16478758	
		

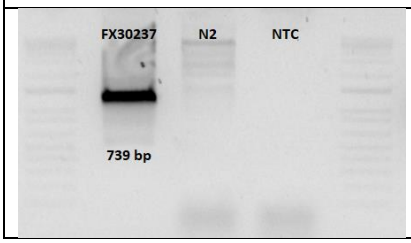
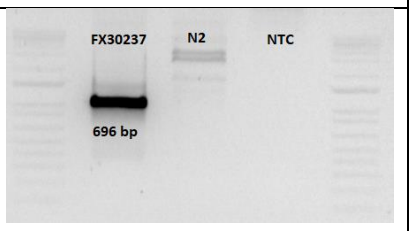
FX30235

tmC20 Inversion2 I:4709755-8256469	*tmC20* Inversion I:114579-8210593
	

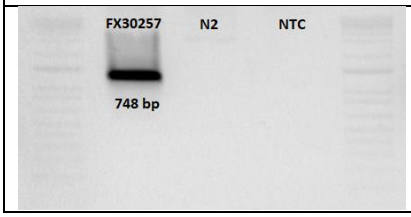
FX30133 / FX30237

Tandem Duplication X: 2932305-2951620	Tandem duplication X: 2939368-2940583
	

FX30237

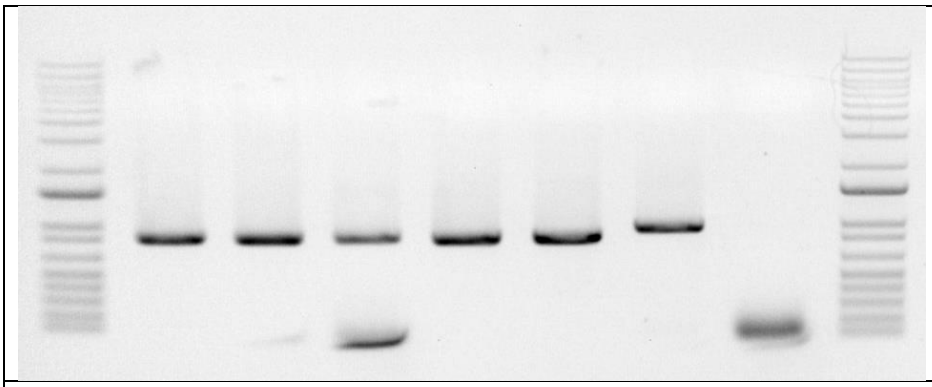
tmC24 Inversion X:8467140-15829281	*tmC24* Inversion X:12460309-15842089
	

FX30257

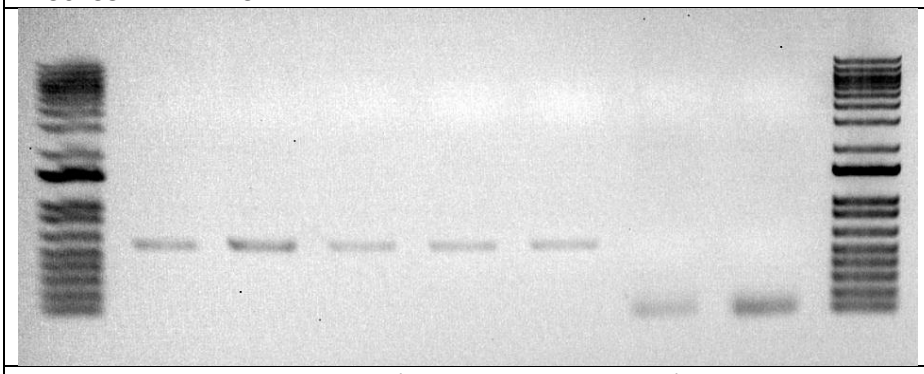
tmC25 Inversion IV: 749886-3418440	*tmC25* Inversion IV: 888732-7203027
	

CRISPR-Cas9

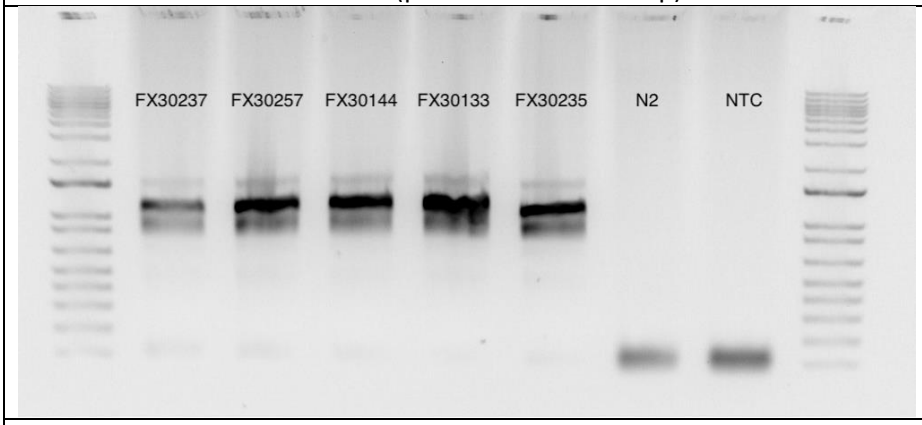
Deletion I:12586101-12586223 (band 857 bp) – Lanes: FX30237 – FX30257 – FX30144 – FX30133 – FX30235 – N2 - NTC



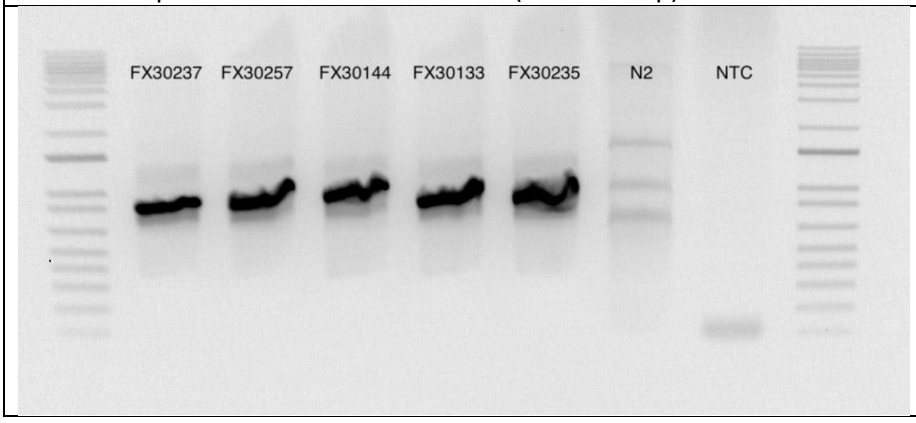
Deletion V:1842989-18465644 (band 567 bp) – Lanes: FX30237 – FX30257 – FX30144 – FX30133 – FX30235 – N2 - NTC



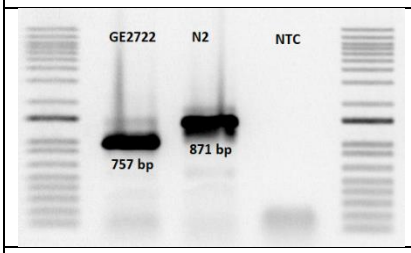
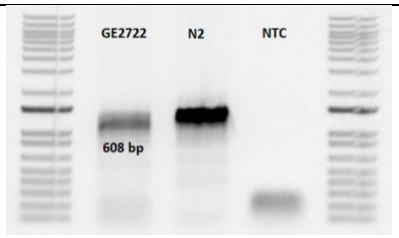
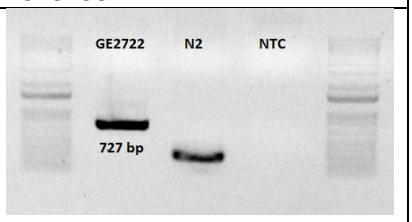
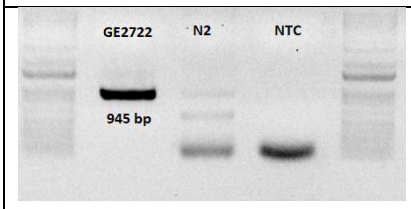
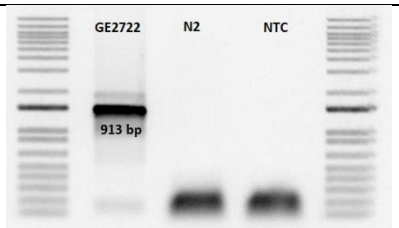
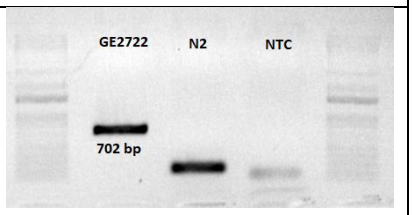
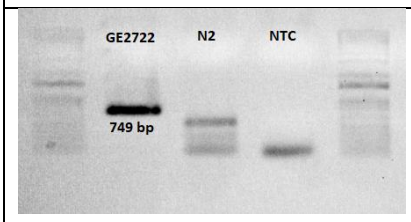
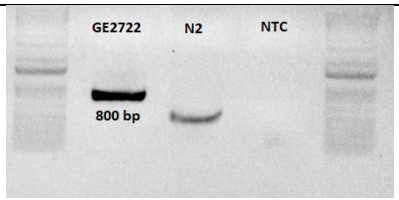
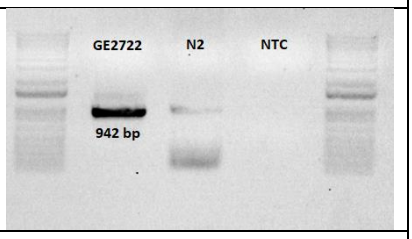
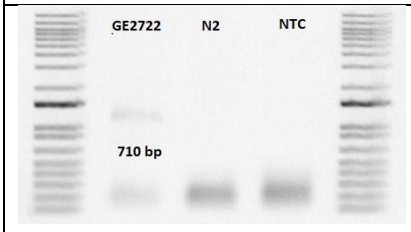
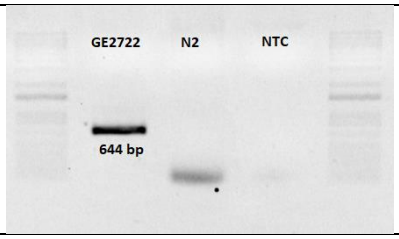
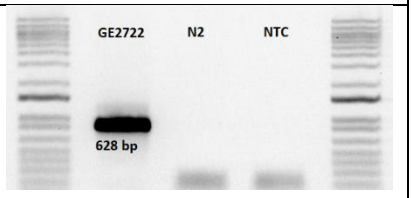
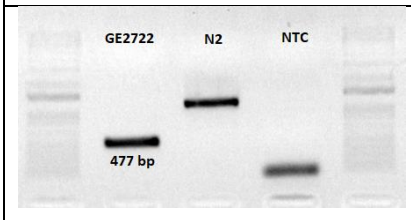
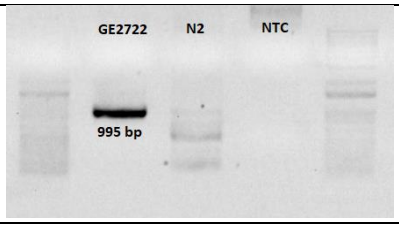
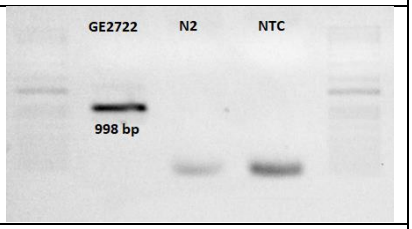
Deletion I:11849276-11841612 (predicted band 685 bp)

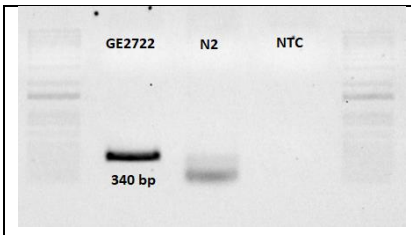


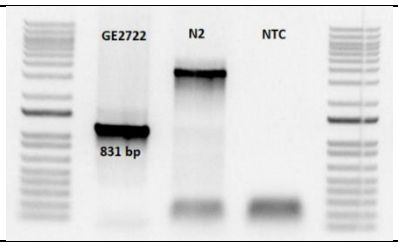
Tandem Duplication V:6236672-6239149 (band 721 bp)

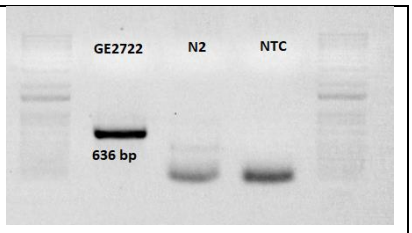


GE2722

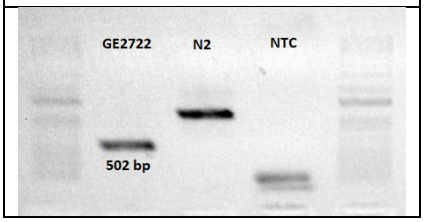
qC1 Deletion III:7265135-7265250	*qC1* Deletion III:4404344-4404470	*qC1* Inversion III:9096426-13737954
		
qC1 Inversion III:6649394-6682735	*qC1* Inversion III:9096455-9955971	*qC1* tandem duplication III:1286508-9955633
		
qC1 deletion III:10407337-10483794	*qC1* tandem duplication III:6645681-6691080	*qC1* tandem duplication III:10407341-10484199
		
qC1 tandem duplication III:9096426-13737954	CGR1 Inversion II:5099553-5099856	Tandem duplication V:4405080-4417827
		
CGR2 Deletion II: 6761122-6761844	CGR3 tandem duplication I:2383609-2389098	CGR1 Tandem duplication II:5096210-5101425
		
CGR2 tandem duplication II:6758980-6763714	Deletion I:11148805-11149851	CGR3 deletion I:2384049-2389112





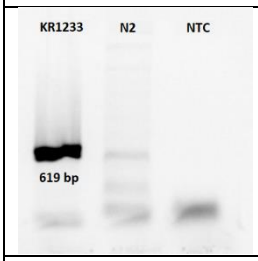


CGR2 deletion II:6760512-6761141

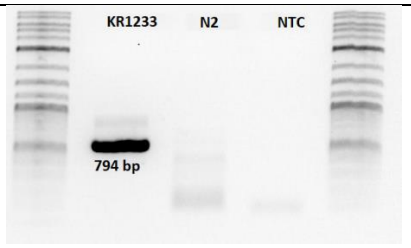


KR1233

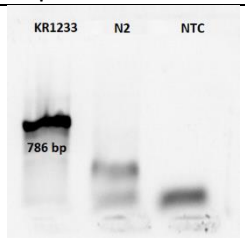
hDf8 I:1966186 I:6412411



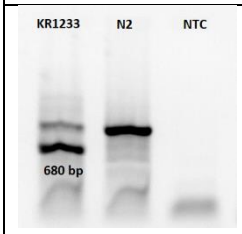
hDf8 Inversion I:6419925-6910540



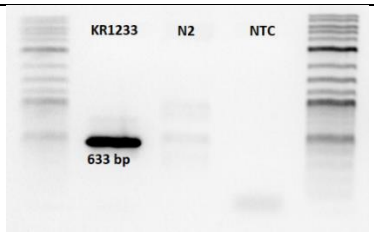
hDf8 Inverted tandem duplication I:1113085-1119781



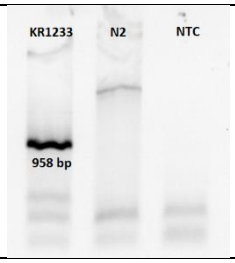
hDf8 Deletion I:1802224-1802589



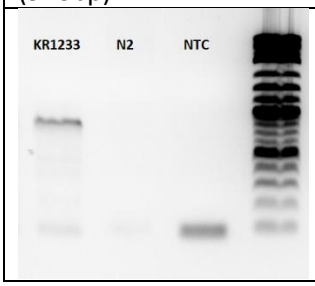
hDf8 Inversion I:2814292-6915195



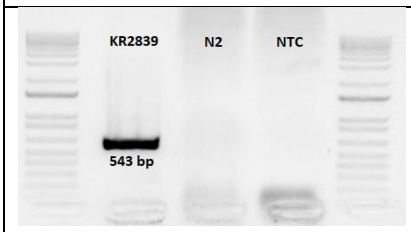
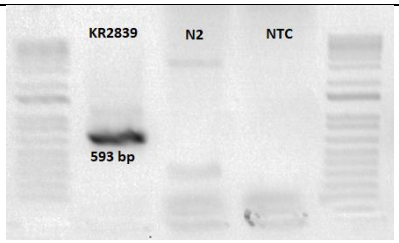
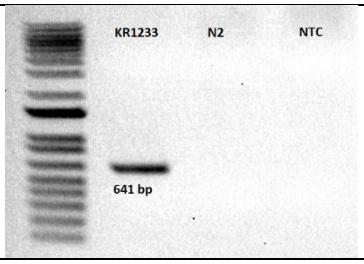
hDf8 Inverted tandem duplication I:6420479-6915192



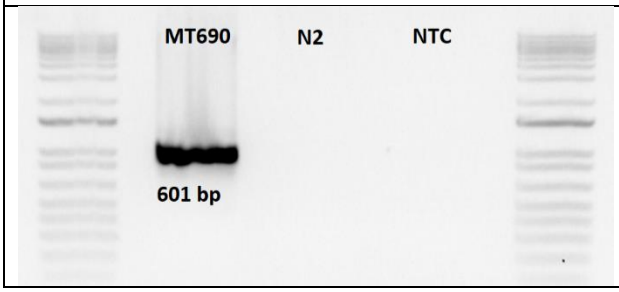
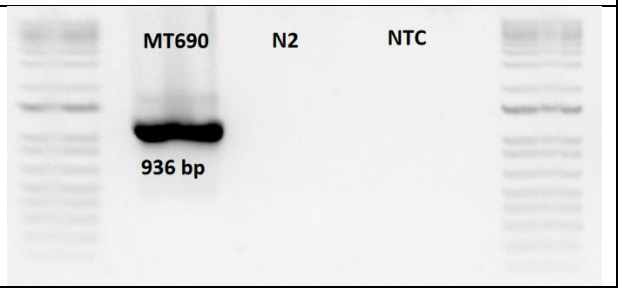
hDf8 Inverted tandem duplication I:3535046-3536437 (916bp)



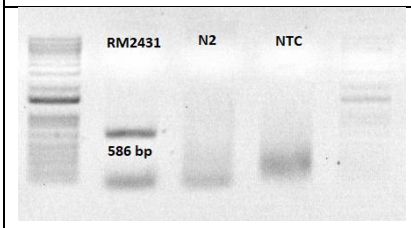
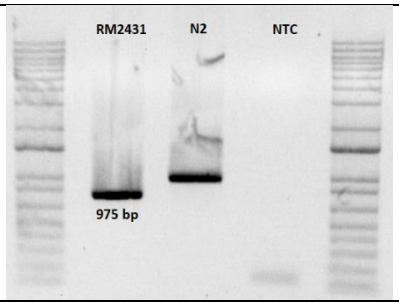
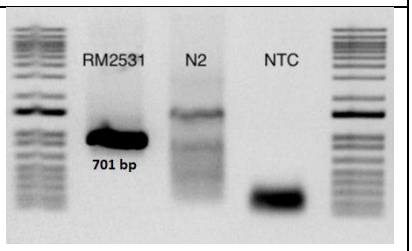
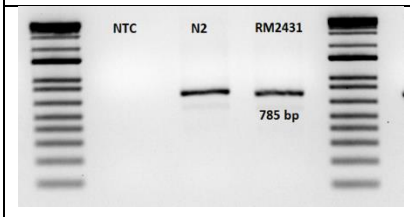
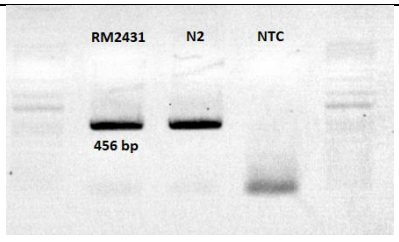
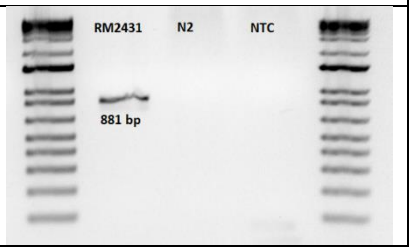
KR2839

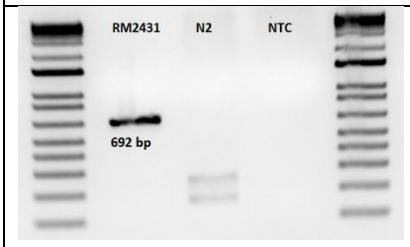
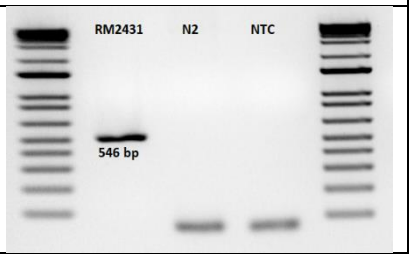
Tandem duplication I:10565986-10577022	*hIn1* Inversion I:11249952-15003739	*hIn1* Tandem duplication I:14573853-14580325
		

MT690

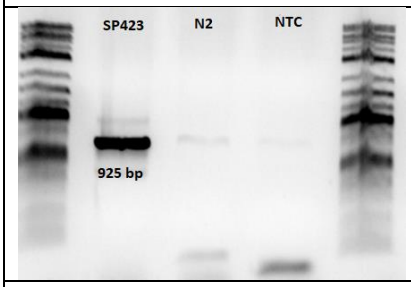
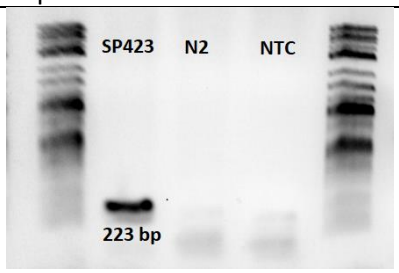
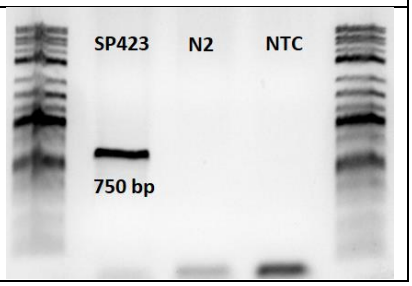
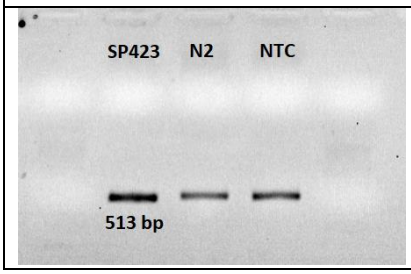
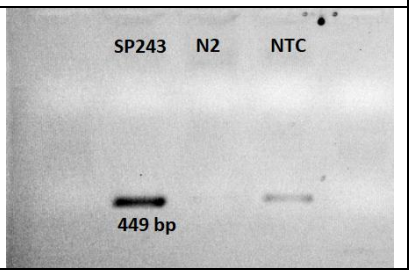
mDf6 Deletion III: 3607207-3695411	Tandem Duplication X:11845164-11852389
	

RM2431

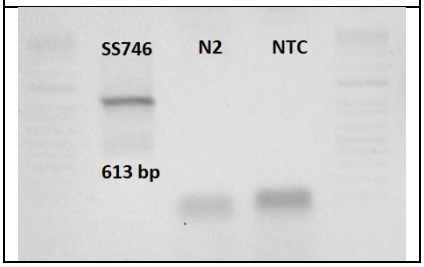
Inverted tandem duplication V:18866326-18866821	Deletion X:7095702-7095968	*hT1* Translocation I:8409987-V:7207631
		
Deletion I:14338214-14338233	**Deletion I:8409987-8410511**	**Inversion: II:8513281-10124446**
		

Tandem Duplication I:1307000-1308796	Tandem duplication IV:12055597-12069818
	

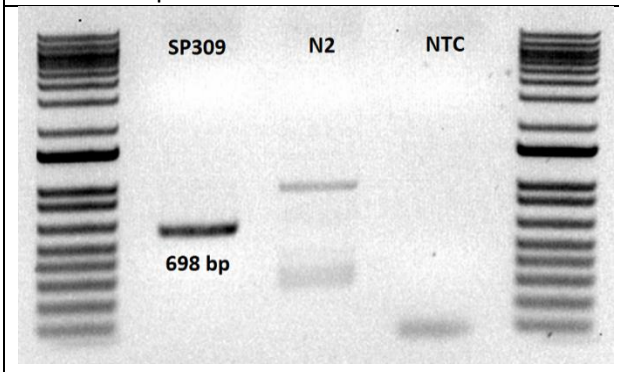
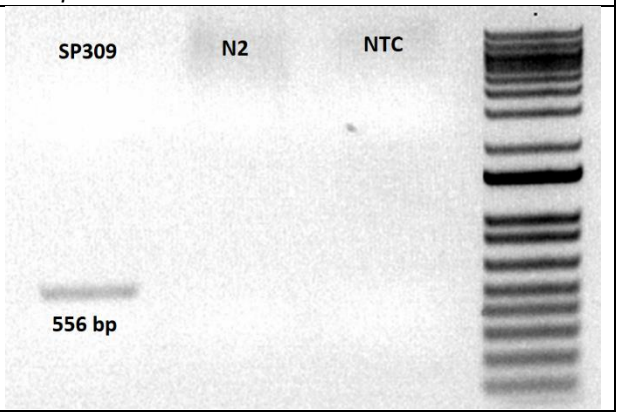
SP423

CGR1 Deletion X:5641151-5647536	CGR1 Inverted tandem duplication X:5638858-5642068	CGR1 Tandem duplication X: 5642328-5652469
		
CGR1 Tandem duplication X:5637751-5642424	Inverted tandem duplication V:16305-17193	
		

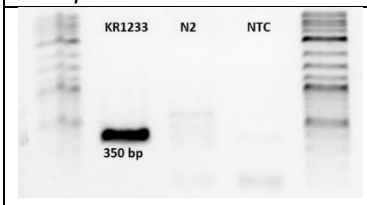
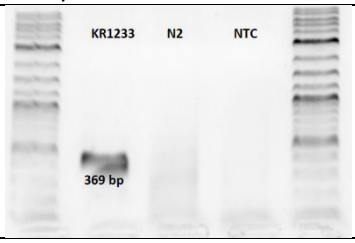
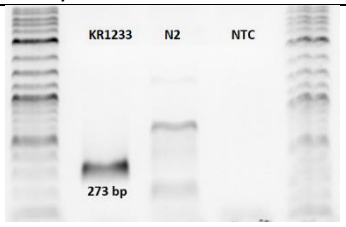
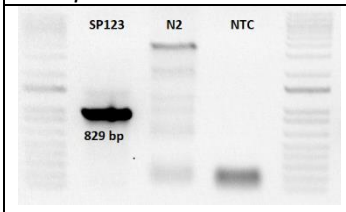
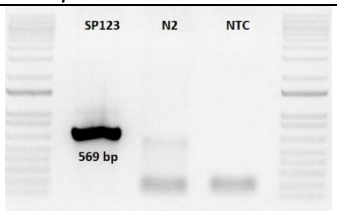
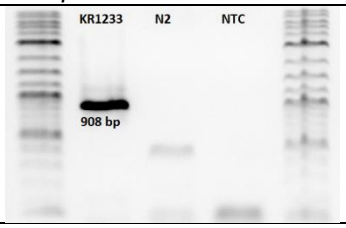
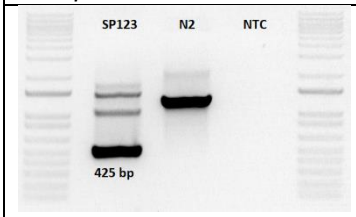
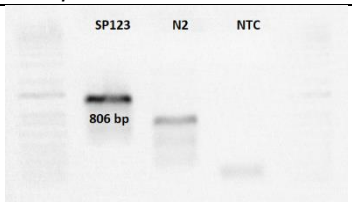
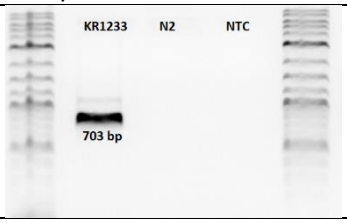
SS746

mT1 Translocation II:6296872-III:3635354


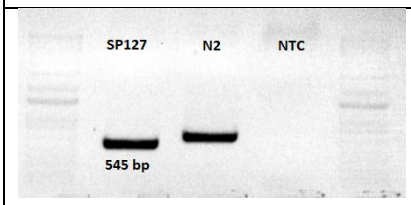
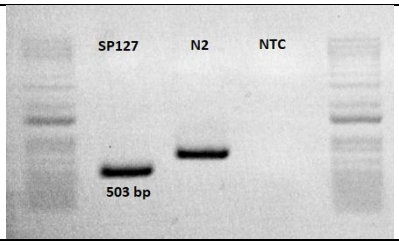
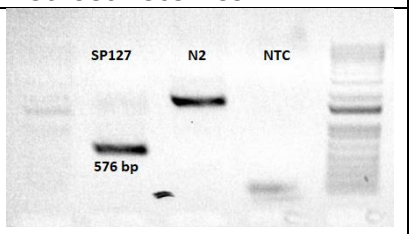
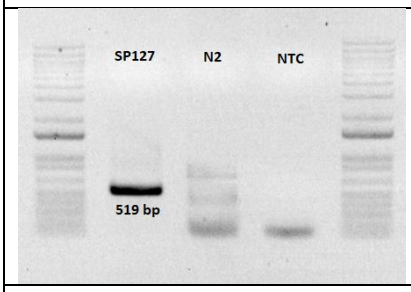
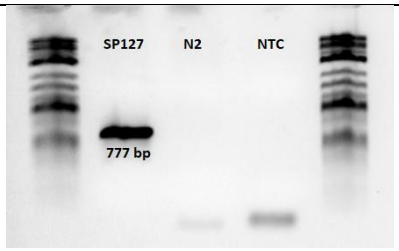
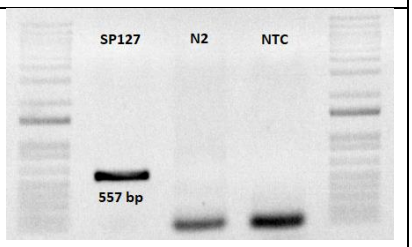
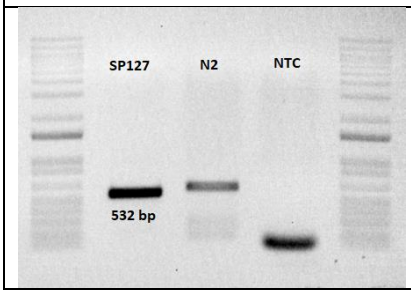
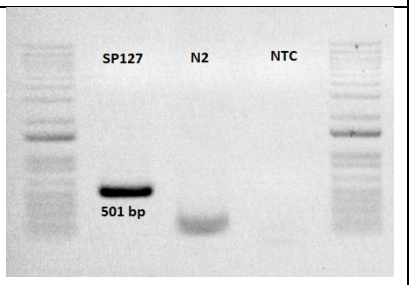
SP309

Tandem Duplication II:3495711-3513352	*mnDp33* Translocation IV:3518557-X:2833653
	

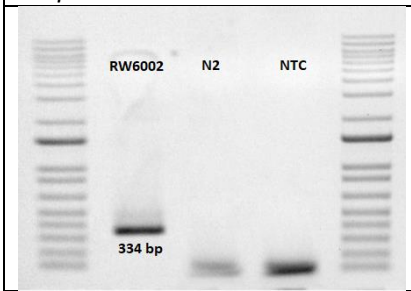
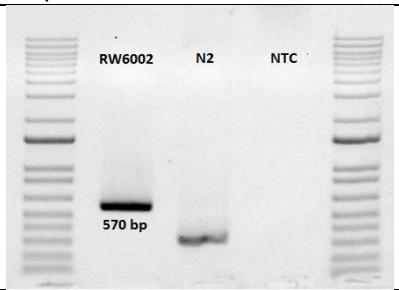
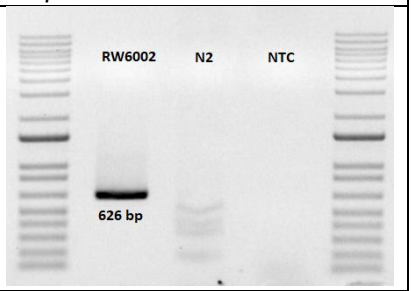
SP123

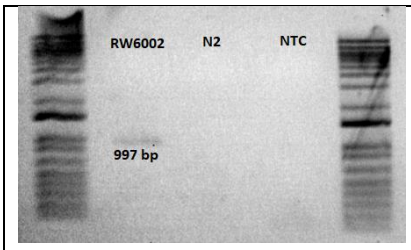
mnDp3 X:14943711-14954885	*mnDp3* X:15422113-15796219	*mnDp3* X:15419295-15421670
		
mnDp3 I:15064452-X:15484795	*mnDp3* X:15437541-15438062	*mnDp3* X:14998958-15009370
		
mnDp3 X:17132213-17133014	*mnDp3* X:15434592-15436210	*mnDp3* X:13175974-13382739
		
mnDp3 X:12974200-12984629		
		

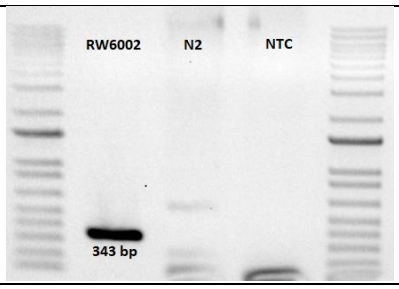
SP127

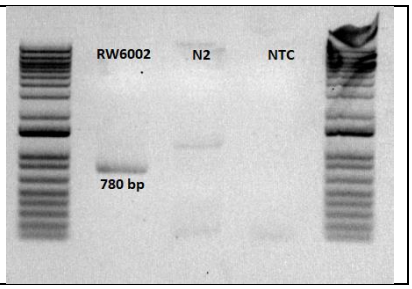
Deletion X:16029231-16029350	Deletion V:4694692-4694952	CGR1 Deletion X:5649867-5651159
		
mnC1 Deletion II:9692491-9695040	CGR1 Inversion X:5646078-5662520	*mnC1* Inversion II:4904692-14909258
		
mnC1 Inversion II:4904677-7535185	*mnC1* Inversion II:7535181-14909251	
		

RW6002

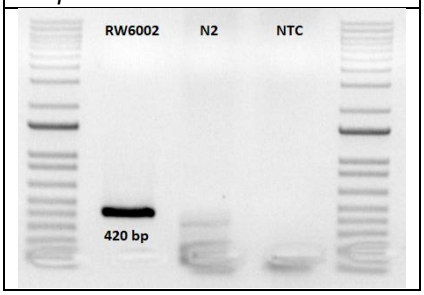
stDp2 X:10301309-II:3907835	*stDp2* X:10301979-10304145	*stDp2* X:10306354-II:3917998
		
Deletion IV:5370371-5371720	Inverted tandem duplication III: 165133-171148	Tandem duplication III:11830810-11834895





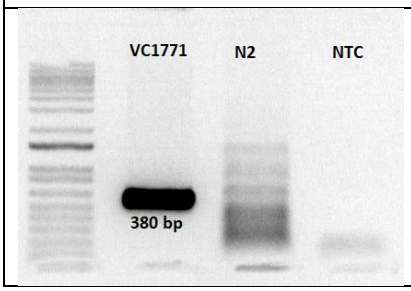


stDp2 X:6841311-10304171

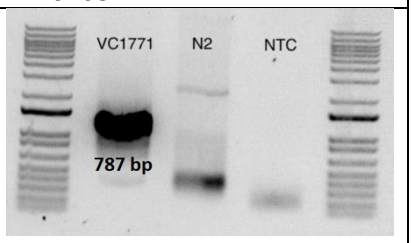


VC1771

Inversion II:11113451-11113761



mIn1 Inversion II:3553628-12704681



Code for Figures with karyoploteR R package

Data are available in github: https://github.com/MTG-Lab/C-elegans-balancers

```
##########################
# map – Figure 1A
 # map
 library("karyoploteR")
 library(biomaRt)
 library(regioneR)
 library("yarrr")
 ######################
 # genes
 data=read.table("mart_export_map.tsv", sep="\t", header=T)
 head(data)
 genes <- toGRanges(data)
 genes
 ####################
 # plot
 png("map.png", width=900, height=900)

 kp <- plotKaryotype(genome = "BSgenome.Celegans.UCSC.ce11", plot.type=2,
 chromosomes=c("chrI", "chrII", "chrIII", "chrIV", "chrV", "chrX"))
 kpAddBaseNumbers(kp, tick.dist=5000000, add.units=TRUE, cex=0.8)

 #-------------------------------------------
 ###chrI
 #hT3
 kpRect(kp, chr="chrI", x0=1, x1=5245743, y0=0.2, y1=0.8, col="grey", r0=0.2, r1=0)
 kpRect(kp, chr="chrX", x0=14773538, x1=17600000, y0=0.2, y1=0.8, col="grey", r0=0.2, r1=0)
 #szT1 - translocation and free duplication
 kpRect(kp, chr="chrI", x0=1, x1=7631470, y0=0.2, y1=0.8,
 col=yarrr::transparent("#FF0000FF", trans.val=.6), r0=0.4, r1=0.2)
 kpRect(kp, chr="chrX", x0=1, x1=2597439, y0=0.2, y1=0.8,
 col=yarrr::transparent("#FF0000FF", trans.val=.6), r0=0.4, r1=0.2)
 kpRect(kp, chr="chrI", x0=1, x1=7454116, y0=0.2, y1=0.8, col="#FF0000FF", r0=0.4, r1=0.2)
 kpRect(kp, chr="chrX", x0=2144605, x1=17600000, y0=0.2, y1=0.8, col="#FF0000FF", r0=0.4,
 r1=0.2)
 #hIn1 - inversion
 kpRect(kp, chr="chrI", x0=11249952, x1=15003739, y0=0.2, y1=0.8,
 col=yarrr::transparent("#B1FF00FF", trans.val=.6), r0=0.2, r1=0)
 kpRect(kp, chr="chrI", x0=11592246, x1=14863482, y0=0.2, y1=0.8, col="#B1FF00FF", r0=0.2,
 r1=0)
```

```
#hDf15 - deletion
kpRect(kp, chr="chrI", x0=12431742, x1=14006044, y0=0.2, y1=0.8,
col=yarrr::transparent("#009BFFFF", trans.val=.6), r0=0.4, r1=0.2)
kpRect(kp, chr="chrI", x0=13300000, x1=13910000, y0=0.2, y1=0.8, col="#009BFFFF", r0=0.4,
r1=0.2)
#hT1 - Translocation
kpRect(kp, chr="chrI", x0=1, x1=8409987, y0=0.2, y1=0.8,
col=yarrr::transparent("slateblue", trans.val=.6), r0=0.6, r1=0.4)
kpRect(kp, chr="chrV", x0=1, x1=7207631, y0=0.2, y1=0.8,
col=yarrr::transparent("slateblue", trans.val=.6) , r0=0.6, r1=0.4)
#kpRect(kp, chr="chrI", x0=1, x1=8409987, y0=0.2, y1=0.8, col="slateblue", r0=0.6, r1=0.4)
kpRect(kp, chr="chrV", x0=1, x1=6515240, y0=0.2, y1=0.8, col="slateblue", r0=0.6, r1=0.4)
#hT2 - translocation
kpRect(kp, chr="chrI", x0=1, x1=13187133, y0=0.2, y1=0.8, col=yarrr::transparent("violet",
trans.val=.6), r0=0.8, r1=0.6)
kpRect(kp, chr="chrIII", x0=4822648, x1=13700000, y0=0.2, y1=0.8,
col=yarrr::transparent("violet", trans.val=.6), r0=0.8, r1=0.6)
kpRect(kp, chr="chrI", x0=1, x1=12513420, y0=0.2, y1=0.8, col="violet", r0=0.8, r1=0.6)
kpRect(kp, chr="chrIII", x0=5107330, x1=13500000, y0=0.2, y1=0.8, col="violet", r0=0.8,
r1=0.6)
#hDf8
kpRect(kp, chr="chrI", x0=1, x1=11034814, y0=0.2, y1=0.8,
col=yarrr::transparent("#009BFFFF", trans.val=.6), r0=1, r1=0.8)
kpRect(kp, chr="chrI", x0=1, x1=6845142, y0=0.2, y1=0.8, col="#009BFFFF", r0=1, r1=0.8)
#-----------------------------------------------
###chrII

#mT1 - translocation
kpRect(kp, chr="chrII", x0=6296872, x1=15000000, y0=0.2, y1=0.8,
col=yarrr::transparent("violetred", trans.val=.6), r0=0.2, r1=0)
kpRect(kp, chr="chrIII", x0=3635354, x1=13700000, y0=0.2, y1=0.8,
col=yarrr::transparent("violetred", trans.val=.6), r0=0.2, r1=0)
kpRect(kp, chr="chrII", x0=6710149, x1=15000000, y0=0.2, y1=0.8, col="violetred", r0=0.2,
r1=0)
kpRect(kp, chr="chrIII", x0=3040846, x1=13500000, y0=0.2, y1=0.8, col="violetred", r0=0.2,
r1=0)
#mIn1 - inversion
kpRect(kp, chr="chrII", x0=3553628, x1=12704681, y0=0.2, y1=0.8,
col=yarrr::transparent("#B1FF00FF", trans.val=.6), r0=0.4, r1=0.2)
kpRect(kp, chr="chrII", x0=3982549, x1=12176069, y0=0.2, y1=0.8, col="#B1FF00FF", r0=0.4,
r1=0.2)
#mnC1 - complex inversion
```

```
kpRect(kp, chr="chrII", x0=4904692, x1=14909258, y0=0.2, y1=0.8,
col=yarrr::transparent("#B1FF00FF", trans.val=.6), r0=0.6, r1=0.4)
kpRect(kp, chr="chrII", x0=6710149, x1=14684456, y0=0.2, y1=0.8, col="#B1FF00FF", r0=0.6,
r1=0.4)
#-------------------------------------------
###chrIII
#nDf6 - deletion
kpRect(kp, chr="chrIII", x0=3607207, x1=3695411, y0=0.2, y1=0.8, col="deepskyblue2", r0=1,
r1=0.8)
#sC1 - complex inversion
kpRect(kp, chr="chrIII", x0=323321, x1=4641137, y0=0.2, y1=0.8,
col=yarrr::transparent("#B1FF00FF", trans.val=.6), r0=0.8, r1=0.6)
kpRect(kp, chr="chrIII", x0=491547, x1=3040846, y0=0.2, y1=0.8, col="#B1FF00FF", r0=0.8,
r1=0.6)
#qC1 - complex inversion
kpRect(kp, chr="chrIII", x0=1286122, x1=13737952, y0=0.2, y1=0.8,
col=yarrr::transparent("#B1FF00FF", trans.val=.6), r0=0.6, r1=0.4)
kpRect(kp, chr="chrIII", x0=2034988, x1=11195201, y0=0.2, y1=0.8, col="#B1FF00FF", r0=0.6,
r1=0.4)
#eT1 - reciprocal translocation
kpRect(kp, chr="chrIII", x0=8200764, x1=13700000, y0=0.2, y1=0.8,
col=yarrr::transparent("purple", trans.val=.6), r0=0.4, r1=0.2)
kpRect(kp, chr="chrV", x0=1, x1=8930675, y0=0.2, y1=0.8, col=yarrr::transparent("purple",
trans.val=.6), r0=0.4, r1=0.2)
kpRect(kp, chr="chrIII", x0=8193843, x1=13700000, y0=0.2, y1=0.8, col="purple", r0=0.4,
r1=0.2)
kpRect(kp, chr="chrV", x0=1, x1=8939684, y0=0.2, y1=0.8, col="purple", r0=0.4, r1=0.2)
#-------------------------------------------
## chrIV
#nT1 - chromoplexy
kpRect(kp, chr="chrIV", x0=1901208, x1=17500000, y0=0.2, y1=0.8,
col=yarrr::transparent("plum", trans.val=.6), r0=0.2, r1=0)
kpRect(kp, chr="chrV", x0=1, x1=16832779, y0=0.2, y1=0.8, col=yarrr::transparent("plum",
trans.val=.6), r0=0.2, r1=0)
kpRect(kp, chr="chrIV", x0=3618259, x1=17500000, y0=0.2, y1=0.8, col="plum", r0=0.2, r1=0)
kpRect(kp, chr="chrV", x0=1, x1=15080152, y0=0.2, y1=0.8, col="plum", r0=0.2, r1=0)
#sDf22 - deletion
kpRect(kp, chr="chrIV", x0=12526000, x1=13141000, y0=0.2, y1=0.8,
col=yarrr::transparent("#009BFFFF", trans.val=.6), r0=0.4, r1=0.2)
kpRect(kp, chr="chrIV", x0=12567720, x1=12788344, y0=0.2, y1=0.8, col="#009BFFFF", r0=0.4,
r1=0.2)
#mnDp33 - Translocated duplication
```

```
kpRect(kp, chr="chrIV", x0=2853000, x1=4028000, y0=0.2, y1=0.8,
col=yarrr::transparent("#FF0000FF", trans.val=.6), r0=0.4, r1=0.2)
kpRect(kp, chr="chrIV", x0=2984263, x1=3958552, y0=0.2, y1=0.8, col="#FF0000FF", r0=0.4,
r1=0.2)
#-------------------------------------------
##chrV
#sC4
kpRect(kp, chr="chrV", x0=16000000, x1=21000000, y0=0.2, y1=0.8,
col=yarrr::transparent("#009BFFFF", trans.val=.6), r0=0.6, r1=0.4)
kpRect(kp, chr="chrV", x0=15071760, x1=21000000, y0=0.2, y1=0.8, col="#009BFFFF", r0=0.6,
r1=0.4)
#eDf43 - deletion + CGR
kpRect(kp, chr="chrV", x0=3466642, x1=3714670, y0=0.2, y1=0.8, col="#009BFFFF", r0=0.8,
r1=0.6)
#e917 - inversion + CGR
kpRect(kp, chr="chrV", x0=4488422, x1=14577013, y0=0.2, y1=0.8, col="#B1FF00FF", r0=0.8,
r1=0.6)
#-------------------------------------------
##chrX
#stDp2
kpRect(kp, chr="chrX", x0=6842000, x1=10309000, y0=0.2, y1=0.8,
col=yarrr::transparent("#FF0000FF", trans.val=.6), r0=0.2, r1=0)
kpRect(kp, chr="chrX", x0=6889674, x1=10133182, y0=0.2, y1=0.8, col="#FF0000FF", r0=0.2,
r1=0)
#mnDp3 - free duplication
kpRect(kp, chr="chrX", x0=12803000, x1=17600000, y0=0.2, y1=0.8,
col=yarrr::transparent("#FF0000FF", trans.val=.6), r0=0.8, r1=0.6)
kpRect(kp, chr="chrX", x0=13201770, x1=17600000, y0=0.2, y1=0.8, col="#FF0000FF", r0=0.8,
r1=0.6)
#mnDp1 - translocated duplication
kpRect(kp, chr="chrX", x0=13794000, x1=17600000, y0=0.2, y1=0.8, col="grey", r0=0.6,
r1=0.4)
kpPlotMarkers(kp, data=genes, labels=genes$value, data.panel=2, cex=0.8,
adjust.label.position = TRUE, text.orientation="vertical", r1=0.5)
dev.off()

####################
####################
# RM2431 – Figure 5E
library("karyoploteR")
library("RColorBrewer")
####################
# data coverage
```

```
data=read.table("RM2431/coverage.tsv", sep="\t", header=1)
mydata=toGRanges(data)
####################
# data SNP
snp=read.table("RM2431/snp.tsv", header=1)
mysnp=toGRanges(snp)
####################
# breakpoints
start=read.table("RM2431/start.tsv", sep="\t")
end=read.table("RM2431/end.tsv", sep="\t")
mystart=toGRanges(start)
myend=toGRanges(end)
####################
# plot
#-----------------------------------
tiff("RM2431.tiff", width=400, height=200)
kp <- plotKaryotype(genome = "BSgenome.Celegans.UCSC.ce11", plot.type=1,
chromosomes=c("chrI", "chrV"))
kpAddBaseNumbers(kp, tick.dist=5000000, add.units=TRUE, cex=0.5)
at <- autotrack(current.track = 1, total.tracks = 2)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel2")[5])
kpAddLabels(kp, labels = "Coverage", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, ymin=0, ymax=2, r0=at$r0, r1=at$r1, side=2, cex=0.6)
kpLines(kp, data = mydata, ymax=2, r0=at$r0, r1=at$r1, col=brewer.pal(n=8, name="Set2")[1])
at <- autotrack(current.track = 2, total.tracks = 2)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel2")[5])
kpAddLabels(kp, labels = "SNP", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, numticks = 2, tick.pos = c(0.5, 1), labels = c("Het", "Hom"), r0=at$r0,
r1=at$r1, side=2, cex=0.6)
kpPoints(kp, data = mysnp, ymax=2, r0=at$r0, r1=at$r1, cex=0.6, col=brewer.pal(n=8,
name="Set2")[1])
kpPlotRegions(kp, mystart, r0=0, r1=0.5, col="#ff8d92")
kpPlotRegions(kp, myend, r0=0, r1=0.5, col="#8d9aff")
kpPlotLinks(kp, data=mystart, data2=myend, col="#fac7ffaa", r0=0.5)
dev.off()
#########################
# CB3475 – Figure 5C
library("karyoploteR")
library("RColorBrewer")
####################
# data coverage
```

```
data=read.table("CB3475_files/coverage.tsv", sep="\t", header=1)
mydata=toGRanges(data)
####################
# data SNP
snp=read.table("CB3475_files/snp.tsv", sep="\t", header=1)
mysnp=toGRanges(snp)
####################
# breakpoints
start=read.table("CB3475_files/start.tsv", sep="\t")
end=read.table("CB3475_files/end.tsv", sep="\t")
mystart=toGRanges(start)
myend=toGRanges(end)
####################
# plot
tiff("CB3475.tiff", width=400, height=200)
kp <- plotKaryotype(genome = "BSgenome.Celegans.UCSC.ce11", plot.type=1,
chromosomes=c("chrI", "chrX"))
kpAddBaseNumbers(kp, tick.dist=5000000, add.units=TRUE, cex=0.5)
at <- autotrack(current.track = 1, total.tracks = 2)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel2")[5])
kpAddLabels(kp, labels = "Coverage", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, ymin=0, ymax=2, r0=at$r0, r1=at$r1, side=2, cex=0.6)
kpLines(kp, data = mydata, ymax=2, r0=at$r0, r1=at$r1, col=brewer.pal(n=8, name="Set2")[1])
at <- autotrack(current.track = 2, total.tracks = 2)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel2")[5])
kpAddLabels(kp, labels = "SNP", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, numticks = 2, tick.pos = c(0.5, 1), labels = c("Het", "Hom"), r0=at$r0,
r1=at$r1, side=2, cex=0.6)
kpPoints(kp, data = mysnp, ymax=2, r0=at$r0, r1=at$r1, cex=0.6, col=brewer.pal(n=8,
name="Set2")[1])
kpPlotRegions(kp, mystart, r0=0, r1=0.5, col="#ff8d92")
kpPlotRegions(kp, myend, r0=0, r1=0.5, col="#8d9aff")
kpPlotLinks(kp, data=mystart, data2=myend, col="#fac7ffaa", r0=0.5)
dev.off()
#########################
#########################
# RW6002 - Figure 4C
library("karyoploteR")
library("RColorBrewer")
####################
# data coverage
data=read.table("RW6002_files/coverage.tsv", sep="\t", header=1)
mydata=toGRanges(data)
```

```
####################
# data SNP
snp=read.table("RW6002_files/snp.tsv", header=1)
mysnp=toGRanges(snp)
####################
# breakpoints
start=read.table("RW6002_files/start.tsv", sep="\t")
end=read.table("RW6002_files/end.tsv", sep="\t")
mystart=toGRanges(start)
myend=toGRanges(end)
####################
# Figure 4c
#----------------------------------
tiff("RW6002.tiff", width=400, height=200)
kp <- plotKaryotype(genome = "BSgenome.Celegans.UCSC.ce11", plot.type=1,
chromosomes=c("chrII","chrX"))
kpAddBaseNumbers(kp, tick.dist=5000000, add.units=TRUE, cex=0.6)
at <- autotrack(current.track = 1, total.tracks = 2)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel2")[5])
kpAddLabels(kp, labels = "Coverage", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, ymin=0, ymax=4, r0=at$r0, r1=at$r1, side=2, cex=0.6)
kpLines(kp, data = mydata, ymax=4, r0=at$r0, r1=at$r1, col=brewer.pal(n=8, name="Set2")[1])
at <- autotrack(current.track = 2, total.tracks = 2)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel2")[5])
kpAddLabels(kp, labels = "SNP", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, numticks = 2, tick.pos = c(0.5, 1), labels = c("Het", "Hom"), r0=at$r0,
r1=at$r1, side=2, cex=0.6)
kpPoints(kp, data = mysnp, ymax=2, r0=at$r0, r1=at$r1, cex=0.6, col=brewer.pal(n=8,
name="Set2")[1])
kpPlotRegions(kp, mystart, r0=0, r1=0.5, col="#ff8d92")
kpPlotRegions(kp, myend, r0=0, r1=0.5, col="#8d9aff")
kpPlotLinks(kp, data=mystart, data2=myend, col="#fac7ffaa", r0=0.5)
#kpDataBackground(kp, data.panel = 2, col=brewer.pal(n=9, name="Set3")[9], r0=0.1, r1=0.3)
dev.off()
#########################
# FX strains – Figure 2A
library("karyoploteR")
library("RColorBrewer")
####################
# data coverage FX30144
data30133=read.table("CRISPR-Cas9_Strains_files/FX30133/coverage.tsv", sep="\t", header=1)
mydata30133=toGRanges(data30133)
```

```
data30144=read.table("CRISPR-Cas9_Strains_files/FX30144/coverage.tsv", sep="\t", header=1)
mydata30144=toGRanges(data30144)
data30235=read.table("CRISPR-Cas9_Strains_files/FX30235/coverage.tsv", sep="\t", header=1)
mydata30235=toGRanges(data30235)
data30237=read.table("CRISPR-Cas9_Strains_files/FX30237/coverage.tsv", sep="\t", header=1)
mydata30237=toGRanges(data30237)
data30257=read.table("CRISPR-Cas9_Strains_files/FX30257/coverage.tsv", sep="\t", header=1)
mydata30257=toGRanges(data30257)
####################
# breakpoints
start30133=read.table("CRISPR-Cas9_Strains_files/FX30133/start.tsv", sep="\t")
end30133=read.table("CRISPR-Cas9_Strains_files/FX30133/end.tsv", sep="\t")
mystart30133=toGRanges(start30133)
myend30133=toGRanges(end30133)
start30235=read.table("CRISPR-Cas9_Strains_files/FX30235/start.tsv", sep="\t")
end30235=read.table("CRISPR-Cas9_Strains_files/FX30235/end.tsv", sep="\t")
mystart30235=toGRanges(start30235)
myend30235=toGRanges(end30235)
start30144=read.table("CRISPR-Cas9_Strains_files/FX30144/start.tsv", sep="\t")
end30144=read.table("CRISPR-Cas9_Strains_files/FX30144/end.tsv", sep="\t")
mystart30144=toGRanges(start30144)
myend30144=toGRanges(end30144)
start30257=read.table("CRISPR-Cas9_Strains_files/FX30257/start.tsv", sep="\t")
end30257=read.table("CRISPR-Cas9_Strains_files/FX30257/end.tsv", sep="\t")
mystart30257=toGRanges(start30257)
myend30257=toGRanges(end30257)
start30237=read.table("CRISPR-Cas9_Strains_files/FX30237/start.tsv", sep="\t")
end30237=read.table("CRISPR-Cas9_Strains_files/FX30237/end.tsv", sep="\t")
mystart30237=toGRanges(start30237)
myend30237=toGRanges(end30237)
startFX=read.table("CRISPR-Cas9_Strains_files/FX_start.tsv", sep="\t")
endFX=read.table("CRISPR-Cas9_Strains_files/FX_end.tsv", sep="\t")
mystartFX=toGRanges(startFX)
myendFX=toGRanges(endFX)
startFX133_237=read.table("CRISPR-Cas9_Strains_files/FX30133_FX30237_start.tsv", sep="\t")
endFX133_237=read.table("CRISPR-Cas9_Strains_files/FX30133_FX30237_end.tsv", sep="\t")
mystartFX133_237=toGRanges(startFX133_237)
myendFX133_237=toGRanges(endFX133_237)
####################
# Figure 2A
#-----------------------------------
png("FXstrains.png", width=900, height=900)
```

```
kp <- plotKaryotype(genome = "BSgenome.Celegans.UCSC.ce11", plot.type=2,
chromosomes=c("chrI", "chrII", "chrIII", "chrIV", "chrV", "chrX"))
kpAddBaseNumbers(kp, tick.dist=5000000, add.units=TRUE, cex=0.5)
at <- autotrack(current.track = 1, total.tracks = 5)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel1")[9])
kpAddLabels(kp, labels = "FX30133", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, ymin=0, ymax=2, r0=at$r0, r1=at$r1, side=2, cex=0.45)
kpLines(kp, data = mydata30133, ymax=2, r0=at$r0, r1=at$r1, col=brewer.pal(n=8,
name="Set2")[1])
at <- autotrack(current.track = 2, total.tracks = 5)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel1")[9])
kpAddLabels(kp, labels = "FX30144", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, ymin=0, ymax=2, r0=at$r0, r1=at$r1, side=2, cex=0.45)
kpLines(kp, data = mydata30144, ymax=2, r0=at$r0, r1=at$r1, col=brewer.pal(n=8,
name="Set2")[2])
at <- autotrack(current.track = 3, total.tracks = 5)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel1")[9])
kpAddLabels(kp, labels = "FX30235", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, ymin=0, ymax=2, r0=at$r0, r1=at$r1, side=2, cex=0.45)
kpLines(kp, data = mydata30235, ymax=2, r0=at$r0, r1=at$r1, col=brewer.pal(n=8,
name="Set2")[3])
at <- autotrack(current.track = 4, total.tracks = 5)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel1")[9])
kpAddLabels(kp, labels = "FX30237", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, ymin=0, ymax=2, r0=at$r0, r1=at$r1, side=2, cex=0.45)
kpLines(kp, data = mydata30237, ymax=2, r0=at$r0, r1=at$r1, col=brewer.pal(n=8,
name="Set2")[4])
at <- autotrack(current.track = 5, total.tracks = 5)
kpDataBackground(kp, r0=at$r0, r1=at$r1, color = brewer.pal(n=8, name="Pastel1")[9])
kpAddLabels(kp, labels = "FX30257", r0=at$r0, r1=at$r1, cex=0.8)
kpAxis(kp, ymin=0, ymax=2, r0=at$r0, r1=at$r1, side=2, cex=0.45)
kpLines(kp, data = mydata30257, ymax=2, r0=at$r0, r1=at$r1, col=brewer.pal(n=8,
name="Set2")[5])
kpPlotLinks(kp, data=mystart30235, data2=myend30235, col=brewer.pal(n=5, name="Set2")[3],
r0=0)
kpPlotLinks(kp, data=mystart30144, data2=myend30144, col=brewer.pal(n=5, name="Set2")[2],
r0=0)
kpPlotLinks(kp, data=mystart30257, data2=myend30257, col=brewer.pal(n=5, name="Set2")[5],
r0=0)
kpPlotLinks(kp, data=mystart30237, data2=myend30237, col=brewer.pal(n=5, name="Set2")[4],
r0=0)
kpPlotLinks(kp, data=mystart30133, data2=myend30133, col=brewer.pal(n=5, name="Set2")[1],
r0=0)
```

```
#kpPlotLinks(kp, data=mystartFX, data2=myendFX, col="yellow", r0=0)
#kpPlotLinks(kp, data=mystartFX133_237, data2=myendFX133_237, col="black", r0=0)
kpDataBackground(kp, data.panel = 2, col=brewer.pal(n=9, name="Set3")[9], r0=0.1, r1=0.3,)
#all
kpPlotRegions(kp, data=c("chrI:11849276-11851612"), col="blue", data.panel=2, r0=0.1,
r1=0.3)
kpPlotRegions(kp, data=c("chrV:18429890-18465644"), col="blue", data.panel=2, r0=0.1,
r1=0.3)
kpPlotRegions(kp, data=c("chrV:6236672-6239149"), col="orange", data.panel=2, r0=0.1,
r1=0.3)
kpPlotRegions(kp, data=c("chrI:12586101-12586223"), col="blue", data.panel=2, r0=0.1,
r1=0.3)
#2
kpPlotRegions(kp, data=c("chrX:2932305-2951620"), col="orange", data.panel=2, r0=0.1,
r1=0.2)
kpPlotRegions(kp, data=c("chrX:2939368-2940583"), col="orange", data.panel=2, r0=0.2,
r1=0.3)
#FX30144
kpPlotRegions(kp, data=c("chrII:5131837-5805620"), col="orange", data.panel=2, r0=0.1,
r1=0.2)
kpPlotRegions(kp, data=c("chrII:5136381-5806749"), col="blue", data.panel=2, r0=0.2,
r1=0.3)
kpPlotRegions(kp, data=c("chrIV:16464405-16465988"), col="blue", data.panel=2, r0=0.1,
r1=0.15)
kpPlotRegions(kp, data=c("chrIV:16464489-16465367"), col="blue", data.panel=2, r0=0.15,
r1=0.2)
kpPlotRegions(kp, data=c("chrIV:16469112-16510351"), col="green", data.panel=2, r0=0.2,
r1=0.25)
kpPlotRegions(kp, data=c("chrIV:16462855-16478758"), col="green", data.panel=2, r0=0.25,
r1=0.3)
#FX30133
kpPlotRegions(kp, data=c("chrV:15826754-15886806"), col="blue", data.panel=2, r0=0.1,
r1=0.3)
#start.regs <- toGRanges(data.frame("chrII", 10e6, 10e6))
#end.regs <- toGRanges(data.frame("chrV", 19e6, 19e6))
#kpPlotLinks(kp, data=start.regs, data2=end.regs, col=brewer.pal(n=5, name="Set2")[8],
data.panel=2, r0=0.1)
dev.off()
```