

Supplemental materials and methods

Plasmids

shRNAs were cloned into pLKO.1-Puro, pLKO.1-GFP, or pLKO.1-mCherry. The latter was generated by amplifying mCherry cDNA with primers containing BamHI and KpnI restriction sites, followed by BamHI and KpnI digestion of PCR product and pLKO.1 backbone and subsequent ligation. sgRNAs were cloned into pLentiCRISPRv2¹. Constructs were verified by Sanger sequencing. shRNA and sgRNA sequences are listed below.

shRNA and sgRNA sequences						
Gene	Species	Type	Library	Identifier	Designation in manuscript	Target sequence
Cdk6	mouse	shRNA	TRC-Mm 1.0 (Mouse)	TRCN0000023153	shCdk6	GCTCAACCCATCGA GAAGTTT
Limk1	mouse	shRNA	TRC-Mm 1.0 (Mouse)	TRCN0000022547	shLimk1	GATGGTGATGAAG GAACTTAT
-	-	shRNA	TRC-Hs 1.0, 1.5, and 2.0 (Human)	SHC002	NTC (non-targeting control)	CAACAAGATGAAGA GCACCAA
CDK6	human	shRNA	TRC-Hs 1.0 (Human)	TRCN0000010081	shCDK6_1	GTCTCACCCATACT TCCAGGA
CDK6	human	shRNA	TRC-Hs 1.0 (Human)	TRCN0000055435	shCDK6_2	TCTGGAGTGTTGG CTGCATAT
LIMK1	human	shRNA	TRC-Hs 1.0 (Human)	TRCN0000199497	shLIMK1_1	GCAACAGGTATCG AGGACTCT
LIMK1	human	shRNA	TRC-Hs 1.0 (Human)	TRCN0000000825	shLIMK1_2	CAGTTGAGCATCTA GGAAGTA
LIMK2	human	shRNA	TRC-Hs 1.0 (Human)	TRCN0000010240	shLIMK2_1	CCCTGAGATGCTG AACGGA
LIMK2	human	shRNA	TRC-Hs 2.0 (Human)	TRCN0000234388	shLIMK2_2	TCGTTCTCTGTGAG ATCATTG
mCherry	-	sgRNA	-	-	sgmCherry	CGCCCTCGATCTC GAACTCG
CDK6	human	sgRNA	GeCKOv2 ¹	HGLibA_08790	sgCDK6_1	TTAGATCGCGATGC ACTACT
CDK6	human	sgRNA	GeCKOv2 ¹	HGLibA_08791	sgCDK6_2	TCTGAACTTCCACG AAAAAG
LIMK1	human	sgRNA	GeCKOv2 ¹	HGLibB_27503	sgLIMK1_1	GTCGCACCAGTACT ATGAGA
LIMK1	human	sgRNA	GeCKOv2 ¹	HGLibB_27504	sgLIMK1_2	CGCTATGGCGAGT CCTGCCA
LIMK1	human	sgRNA	GeCKOv2 ¹	HGLibA_34829	sgLIMK1_3	TGACGGGGACACC TACACGC

Production of lentiviruses

Lentiviral particles were harvested 48 and 72 hours after transfection of HEK293T cells using TransIT-LT1 (Mirus, Madison, Wisconsin, USA) with a lentiviral expression plasmid (pLKO.1-Puro, pLKO.1-GFP, pLKO.1-mCherry, pLentiCRISPRv2) and packaging plasmids pMD2.G and psPAX2.

Lentiviral transduction of AML cells

AML cell lines were transduced with lentiviral particles at a density of 5×10^5 cells/mL. For transduction of PDX cells, lentiviruses were centrifuged onto RetroNectin (Takara, Kusatsu, Japan)-coated plates for 45 minutes at 4 200 rpm. Cells were carefully added and centrifuged onto the plate for 5 minutes at 1 200 rpm. Polybrene (Merck Millipore) was added to a final concentration of 10 μ g/mL. Virus was removed 24 hours after transduction by washing cells three times with PBS. Puromycin (Sigma Aldrich, St. Louis, Missouri, USA) selection was initiated 30 hours after transduction (2 μ g/mL for AML cell lines, 1 μ g/mL for PDX samples) if applicable. Transformed murine bone marrow cells were transduced with pLKO.1-GFP and pLKO.1-mCherry constructs encoding the same shRNA sequence via spininfection as described above. Virus was removed 24 hours later by washing cells three times with PBS. Viable, highly GFP/mCherry double-positive cells were sorted 48 hours after transduction with a FACSAria I (BD Biosciences, San Jose, California, USA).

Proliferation assessment by manual cell counting

Cells were counted with trypan blue staining in a Neubauer counting chamber and seeded at 1×10^5 /mL. Cells were counted on days 2, 4, and 7 after seeding and subcultured if required. Proliferation constants were calculated by exponential growth modeling in GraphPad Prism 6 (GraphPad Software, San Diego, California, USA).

Competition assay

Cells were transduced with pLKO.1-GFP lentiviruses at transduction efficiencies between 20 and 60%. Virus was removed after 24 hours. The proportion of GFP-positive cells was analyzed on days 3, 6, 10, 13, and 17 post-transduction with a LSR Fortessa flow cytometer (BD Biosciences). After normalization to day 3, log-transformed data for the final day of the experiment were tested for significance by unpaired t-tests (NTC vs. shLIMK1).

Colony formation assays

Human AML cell lines (1.1×10^3 cells per dish) were plated in methylcellulose medium (HSC002SF, R&D Systems, Minneapolis, Minnesota, USA) in technical duplicates and counted after nine or ten days as indicated. PDX cells (AML-393, 2×10^3 cells per dish; AML-

388, 3×10^3 cells per dish) were plated in methylcellulose medium (MethoCult H4534 Classic without EPO, Stem Cell Technologies; supplemented with 10 ng/mL human FLT3 ligand and 10 ng/mL human thrombopoietin (PeproTech)) in technical duplicates and counted after eleven days. Tile scans were taken with a Cell Observer microscope (Zeiss, Oberkochen, Germany) at 2.5 x magnification.

EdU incorporation assay

EdU incorporation and staining were performed as described in the manufacturer's protocol (Click-iT Plus EdU Alexa Fluor 647 Flow Cytometry Assay Kit, Invitrogen, Carlsbad, California, USA). Briefly, cells were incubated for eight hours with 5 μ M EdU seven days post transduction, fixed in the provided fixation buffer, and stored at 4 °C until staining and analyzed with an LSR Fortessa flow cytometer (BD Biosciences).

CFSE tracking

1×10^6 cells were labeled with 10 μ M CFSE (CFSE Cell Division Tracker Kit, BioLegend, San Diego, California, USA) for 20 minutes at 37 °C. Fifty % of the sample were fixed in 4 % paraformaldehyde in PBS and stored at 4 °C (baseline), whereas the remaining 50 % were cultured for four days, then fixed in 4 % paraformaldehyde in PBS and analyzed with an LSR Fortessa flow cytometer (BD Biosciences). Proliferation indices were calculated using ModFit LT 3.3 software (Verity Software House, Topsham, Maine, USA).

Analysis of cell morphology

5×10^4 to 1×10^5 cells were washed once with PBS and applied to microscopy slides using a Shandon Cytospin 3 centrifuge (Thermo Fisher Scientific). Slides were stained with modified May-Grünwald and Giemsa staining solutions (Sigma Aldrich) and mounted with Entellan Neu (Merck Millipore). Pictures were taken with a Cell Observer microscope (Zeiss) at 40 x magnification.

Protein analysis

Protein lysates were generated as previously described². SDS-PAGE and western blotting were performed using standard procedures. Membranes were probed with rabbit anti-LIMK1 (Cell Signaling Technology, Danvers, Massachusetts, USA; #3842), rabbit anti-phospho-cofilin (Cell Signaling Technology, #3313), rabbit anti-cofilin (Cell Signaling Technology, #5175), mouse anti-LIMK2 (Santa Cruz Biotechnology, Dallas, Texas, USA; #sc-365414), mouse anti-GAPDH (Santa Cruz Biotechnology, #sc-25778), mouse anti-CDK6 (Cell Signaling Technology, #3136), rabbit anti-HSP90 (Santa Cruz Biotechnology, #sc-7947), rabbit anti-phospho-Rb (Cell Signaling Technology, #9308), rabbit anti-PARP (Cell Signaling Technology,

#9542), or mouse anti-Rb (Cell Signaling Technology, 9309) antibodies. Detection was performed using HRP-conjugated (GE Healthcare, Chicago, Illinois, USA) or near-infrared-labeled (LI-COR Biosciences, Lincoln, Nebraska, USA) secondary antibodies.

RNA isolation, cDNA synthesis, and qRT-PCR

RNA was isolated using the RNeasy Micro Kit (qRT-PCR) or RNeasy Mini Plus Kit (RNA-seq) (Qiagen, Hilden, Germany). cDNA synthesis of 0.1-1 µg total RNA was performed using the High Capacity cDNA RT Kit (Applied Biosystems, Foster City, California, USA). SYBR Green qRT-PCRs were run on a C1000 Touch Thermal Cycler (BioRad, Hercules, California, USA) with iTaq Universal SYBR Green Supermix (BioRad). Results were analyzed using the $\Delta\Delta CT$ method. Primer sequences are listed below.

Primer sequences for qRT-PCR			
Gene	Species	Forward primer (5'-3')	Reverse primer (5'-3')
Cdk6	mouse	AGAAGTCCTGCTCCAGTCCA	CACGTCTGAACTTCCACGAA
Limk1	mouse	TGGACGAGATCGACTTGCTG	GCAGCTCCTCAAGACAGGTT
Limk2	mouse	TCTGAGGCGCAGTAACAGC	CTGAGCGGCTTATGTCCCG
Gapdh	mouse	AGGTCGGTGTGAACGGATTGG	TGTAGACCATGTAGTTGAGGTCA

Multivariable regression analysis of TCGA data

Multivariable regression analysis with backward variable selection was performed on TCGA samples with available RNA-seq and survival data (n=132) using the rms package³ in R 3.5.1. The following variables were included in building the model: *LIMK1* expression [$\log_2(\text{TPM})$] as dependent variable and cytogenetics (normal karyotype, other) as well as mutation status of *FLT3*, *NPM1*, *DNMT3A*, *IDH1*, *IDH2*, *TET2*, *NRAS*, *KRAS*, *CEBPA*, and *RUNX1* as independent variables. Genes considered for analysis were mutated in at least six patients (range, 6-35). P values < 0.05 were considered significant.

RNA sequencing

Libraries were generated using the TruSeq Stranded Total RNA Kit from Illumina (San Diego, California, USA). Eight samples were multiplexed per lane. Paired-end sequencing was performed on an Illumina HiSeq 4000 device with a fragment read length of 100 bp. $8.6\text{-}10.2 \times 10^7$ reads were obtained per sample with a mapping rate of 0.94-0.95. Reads were aligned to the hg19_GRCh37 human reference genome using STAR⁴ version 2.5.2b with SAMtools version 0.1.19. Read summarization was done using the featureCounts⁵ function of Subread⁶ version 1.5.1. Further details are available from <https://github.com/DKFZ-ODCF/RNAseqWorkflow>.

The data discussed in this publication have been deposited in NCBI's Gene Expression Omnibus⁷ and are accessible through GEO Series accession number GSE150857 (<https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE150857>).

R code used for differential gene expression analysis of LIMK1 knockdown experiments

```
library(DESeq2) # Main package
library(vsn) # meanSdPlot

# import summarized feature count table & coldata table
input <- read.csv("Input_LIMK1_reverseReads.csv", sep = ";",
                 stringsAsFactors = FALSE, row.names = "gene_id",
                 check.names = FALSE)
coldata <- read.csv("coldata_LIMK1.csv", header = TRUE, sep = ";",
                  row.names = 1)
coldata$shRNA <- as.factor(coldata$shRNA)
coldata$sum.shRNA <- as.factor(coldata$sum.shRNA)
coldata$Replicate <- as.factor(coldata$Replicate)

# remove gene names and move to separate data frame
gene.names <- read.csv("Input_LIMK1_reverseReads.csv", sep = ";",
                     stringsAsFactors = FALSE)
gene.names <- gene.names[, 1:2]
colnames(input) <- c("name", "Scr_DMSO_Rep1", "Scr_DMSO_Rep2",
                  "635_DMSO_Rep1",
                  "635_DMSO_Rep2", "825_DMSO_Rep1", "825_DMSO_Rep2")
cts <- input[, -1]
all(rownames(coldata) == colnames(cts))

# make DESeq2 object
dds.LIMK1 <- DESeqDataSetFromMatrix(
  countData = cts,
  colData = coldata,
  design = ~shRNA
)
dds.LIMK1

rld <- rlog(dds.LIMK1)
plotPCA(rld,
        intgroup = "shRNA",
        returnData = FALSE
)

# Problem: There seems to be a batch effect based on replicate
# --> include replicate number in coldata and account for in the design
formula
deseq_vst <- rlog(dds.LIMK1, blind = FALSE)
deseq_batch.corr <- limma::removeBatchEffect(assay(deseq_vst),
deseq_vst$Replicate)
deseq_pca.batch.corr <- prcomp(t(deseq_batch.corr), rank. = 2)
deseq_pca_batch <- prcomp(t(assay(deseq_vst)), rank. = 2)
summary(deseq_pca_batch)
summary(deseq_pca.batch.corr)

library(magrittr)
library(dplyr)
library(ggplot2)
```

```

library(ggrepel)
library(tidyr)
library(ggthemes)

plot_pca_batch <- deseq_pca_batch$x %>%
  as_tibble() %>%
  cbind(coldata) %>%
  ggplot() +
  geom_point(aes(x = PC1, y = PC2, color = shRNA), size = 3) +
  geom_text_repel(aes(x = PC1, y = PC2, label = Replicate), size = 3) +
  scale_x_continuous("PC1 (43%)") +
  scale_y_continuous("PC2 (32%)") +
  theme_few() +
  labs(
    title = "Principal Component Analysis",
    subtitle = "With batch effects"
  )
plot_pca_batch

plot_pca_batch.corr <- deseq_pca.batch.corr$x %>%
  as_tibble() %>%
  cbind(coldata) %>%
  ggplot() +
  geom_point(aes(x = PC1, y = PC2, color = shRNA), size = 3) +
  geom_text_repel(aes(x = PC1, y = PC2, label = Replicate), size = 3) +
  scale_x_continuous("PC1 (47%)") +
  scale_y_continuous("PC2 (32%)") +
  theme_few() +
  labs(
    title = "Principal Component Analysis",
    subtitle = "Without batch effect"
  )
plot_pca_batch.corr

# Batch correction by relicate improves PCA --> include in analysis design
formula

# Pre-filtering by TPM -> TPM of 0.5 as cutoff for each sample -> total of
6x 0.5 = 3
reverse.reads.TPM <- read.csv("Input_LIMK1_reverseReads_TPM.csv", sep =
";",
                             stringsAsFactors = FALSE, row.names =
"gene_id", check.names = FALSE)
reverse.reads.TPM_0.5 <- reverse.reads.TPM[rowSums(reverse.reads.TPM[,
2:7]) > 3, ]
TPM.filtered.genes <- as.data.frame(reverse.reads.TPM_0.5$name, row.names =
rownames
                                (reverse.reads.TPM_0.5),
                                stringsAsFactors = FALSE)
colnames(TPM.filtered.genes) <- "name"
TPM.filtered.genes$gene_id <- rownames(TPM.filtered.genes)
input$gene_id <- rownames(input)

input_TPM_filtered <- merge(input, TPM.filtered.genes, by = c("gene_id",
"name"))
rownames(input_TPM_filtered) <- input_TPM_filtered$gene_id

cts <- input_TPM_filtered[, -(1:2)]

# Make new dds object
dds.LIMK1.batch.corr <- DESeqDataSetFromMatrix(
  countData = cts,
  colData = coldata,

```

```

    design = ~ Replicate + shRNA
  )
dds.LIMK1.batch.corr

# set reference level
dds.LIMK1.batch.corr$shRNA <- relevel(dds.LIMK1.batch.corr$shRNA, ref =
"Scr")

# Run analysis
dds.LIMK1.batch.corr <- DESeq(dds.LIMK1.batch.corr)

# QC
plotDispEsts(dds.LIMK1.batch.corr)

# Export normalized read counts
dds.LIMK1.batch.corr <- estimateSizeFactors(dds.LIMK1.batch.corr)
Normalized_read_count <- as.data.frame(counts(dds.LIMK1.batch.corr,
normalized = TRUE))
Normalized_read_count$gene_id <- row.names(Normalized_read_count)
Normalized_read_count <- merge(Normalized_read_count, gene.names, by =
"gene_id")
Normalized_read_count <- Normalized_read_count[, c(1, 8, 2:7)]

# call results
resultsNames(dds.LIMK1.batch.corr)
# "Intercept"      "replicate_2_vs_1"      "shRNA_LIMK1_1_vs_Scr"
"shRNA_LIMK1_2_vs_Scr"

results.LIMK1_batch.corr_sh1_0.05 <- results(dds.LIMK1.batch.corr,
name = "shRNA_LIMK1_1_vs_Scr",
alpha = 0.05)
results.LIMK1_batch.corr_sh2_0.05 <- results(dds.LIMK1.batch.corr,
name = "shRNA_LIMK1_2_vs_Scr",
alpha = 0.05)
summary(results.LIMK1_batch.corr_sh1_0.05)
summary(results.LIMK1_batch.corr_sh2_0.05)

# many low counts -> use IHW package for filtering
library(IHW)
results.LIMK1_batch.corr_sh1_0.05 <- results(dds.LIMK1.batch.corr,
name = "shRNA_LIMK1_1_vs_Scr",
filterFun = IHW::ihw, alpha =
0.05)
results.LIMK1_batch.corr_sh2_0.05 <- results(dds.LIMK1.batch.corr,
name = "shRNA_LIMK1_2_vs_Scr",
filterFun = IHW::ihw, alpha =
0.05)
summary(results.LIMK1_batch.corr_sh1_0.05)
summary(results.LIMK1_batch.corr_sh2_0.05)

# MA plots -> Detach limma, otherwise limma's MA plot function is used
detach("package:limma", unload = TRUE)

plotMA(results.LIMK1_batch.corr_sh1_0.05, ylim = c(-6, 6))
plotMA(results.LIMK1_batch.corr_sh2_0.05, ylim = c(-6, 6))

# Apply LFC shrinkage and plot again
results.LIMK1_batch.corr_sh1_0.05_LFC <- lfcShrink(dds.LIMK1.batch.corr,
res = results.LIMK1_batch.corr_sh1_0.05,
coef = "shRNA_LIMK1_1_vs_Scr", type = "apeglm"
)

results.LIMK1_batch.corr_sh2_0.05_LFC <- lfcShrink(dds.LIMK1.batch.corr,

```

```

    res = results.LIMK1_batch.corr_sh2_0.05,
    coef = "shRNA_LIMK1_2_vs_Scr", type = "apeglm"
)
summary(results.LIMK1_batch.corr_sh1_0.05_LFC)
summary(results.LIMK1_batch.corr_sh2_0.05_LFC)

plotMA(results.LIMK1_batch.corr_sh1_0.05_LFC, ylim = c(-4, 4),
        main = "LIMK1_1, LFC shrinkage = apeglm")
plotMA(results.LIMK1_batch.corr_sh2_0.05_LFC, ylim = c(-5, 5),
        main = "LIMK1_2, LFC shrinkage = apeglm")

# Add gene symbols and export all data
dds.LIMK1_1.vs.Scr.all.genes <-
as.data.frame(results.LIMK1_batch.corr_sh1_0.05_LFC,
              row.names = NULL)

dds.LIMK1_1.vs.Scr.all.genes$gene_id <-
rownames(dds.LIMK1_1.vs.Scr.all.genes)
dds.LIMK1_1.vs.Scr.all.genes <- merge(dds.LIMK1_1.vs.Scr.all.genes,
gene.names,
                                   by = "gene_id")
dds.LIMK1_1.vs.Scr.all.genes <- dds.LIMK1_1.vs.Scr.all.genes[, c(1, 7,
2:6)]
dds.LIMK1_1.vs.Scr.all.genes <- dds.LIMK1_1.vs.Scr.all.genes[order(
  dds.LIMK1_1.vs.Scr.all.genes$log2FoldChange,
  dds.LIMK1_1.vs.Scr.all.genes$padj
), ]

dds.LIMK1_2.vs.Scr.all.genes <-
as.data.frame(results.LIMK1_batch.corr_sh2_0.05_LFC,
              row.names = NULL)

dds.LIMK1_2.vs.Scr.all.genes$gene_id <-
rownames(dds.LIMK1_2.vs.Scr.all.genes)
dds.LIMK1_2.vs.Scr.all.genes <- merge(dds.LIMK1_2.vs.Scr.all.genes,
gene.names,
                                   by = "gene_id")
dds.LIMK1_2.vs.Scr.all.genes <- dds.LIMK1_2.vs.Scr.all.genes[, c(1, 7,
2:6)]
dds.LIMK1_2.vs.Scr.all.genes <- dds.LIMK1_2.vs.Scr.all.genes[order(
  dds.LIMK1_2.vs.Scr.all.genes$log2FoldChange,
  dds.LIMK1_2.vs.Scr.all.genes$padj
), ]
LIMK1_all_genes <- merge(dds.LIMK1_1.vs.Scr.all.genes,
dds.LIMK1_2.vs.Scr.all.genes,
                        by = c("gene_id", "name", "baseMean"))

# Extract significant genes and their overlap between both shRNAs
LIMK1_1.significant_0.05 <- dds.LIMK1_1.vs.Scr.all.genes[
  dds.LIMK1_1.vs.Scr.all.genes$padj < 0.05, ]
LIMK1_2.significant_0.05 <- dds.LIMK1_2.vs.Scr.all.genes[
  dds.LIMK1_2.vs.Scr.all.genes$padj < 0.05, ]
significant_0.05_both.shRNAs <- merge(LIMK1_1.significant_0.05,
LIMK1_2.significant_0.05, by = c("gene_id", "name", "baseMean"))

# Remove shRNAs that are regulated in opposite direction by the two shRNAs
significant_0.05_both.shRNAs_ <- significant_0.05_both.shRNAs[
  (significant_0.05_both.shRNAs$log2FoldChange.x < 0 &
  significant_0.05_both.shRNAs$log2FoldChange.y < 0) |
  (significant_0.05_both.shRNAs$log2FoldChange.x > 0 &
  significant_0.05_both.shRNAs$log2FoldChange.y > 0), ]

# Extract normalized read count of significant genes

```

```

significant_0.05_both.shRNAs_readCounts <-
  merge(significant_0.05_both.shRNAs, Normalized_read_count, by =
    c("gene_id", "name"))
significant_0.05_both.shRNAs_readCounts$logfc_635_Rep1 <- log2(
  significant_0.05_both.shRNAs_readCounts$`635_DMSO_Rep1` /
  significant_0.05_both.shRNAs_readCounts$Scr_DMSO_Rep1)
significant_0.05_both.shRNAs_readCounts$logfc_635_Rep2 <- log2(
  significant_0.05_both.shRNAs_readCounts$`635_DMSO_Rep2` /
  significant_0.05_both.shRNAs_readCounts$Scr_DMSO_Rep2)
significant_0.05_both.shRNAs_readCounts$logfc_825_Rep1 <- log2(
  significant_0.05_both.shRNAs_readCounts$`825_DMSO_Rep1` /
  significant_0.05_both.shRNAs_readCounts$Scr_DMSO_Rep1)
significant_0.05_both.shRNAs_readCounts$logfc_825_Rep2 <- log2(
  significant_0.05_both.shRNAs_readCounts$`825_DMSO_Rep2` /
  significant_0.05_both.shRNAs_readCounts$Scr_DMSO_Rep1)
significant_0.05_both.shRNAs_readCounts$mean.logfc <- rowMeans(
  significant_0.05_both.shRNAs_readCounts[, 18:21])
significant_0.05_both.shRNAs_readCounts <-
significant_0.05_both.shRNAs_readCounts[
  order(significant_0.05_both.shRNAs_readCounts$mean.logfc), ]

## Annotate gene categories to filter for protein coding genes for GSEA

# Remove version info from Ensembl gene IDs
library(stringi)
gene_ids_all <- unlist(
  stri_split(
    str = rownames(dds.LIMK1.batch.corr),
    regex = "\\\\.\\.\\d*", # pattern is dot (.) followed by some number
    omit_empty = TRUE
  )
)

head(gene_ids_all) # new variable which hasn't been assigned yet to a
matrix

# Map Ensembl gene IDs to gene names
library(biomaRt)
mart <- useEnsembl(
  biomart = "ensembl",
  dataset = "hsapiens_gene_ensembl",
  verbose = TRUE,
  mirror = "uswest"
)

mart

head(listAttributes(mart))

annotations <- getBM(
  attributes = c("ensembl_gene_id", "external_gene_name", "gene_biotype"),
  filters = "ensembl_gene_id",
  values = gene_ids_all,
  mart = mart
)

colnames(annotations)[2] <- "name"

# Filter for protein coding genes
LIMK1_all_genes$ensembl_gene_id <- unlist(
  stri_split(

```

```

    str = LIMK1_all_genes$gene_id,
    regex = "\\\\.\\d*", # pattern is dot (.) followed by some number
    omit_empty = TRUE
  )
)

# Use mart annotation for updated gene symbols
Normalized_read_count$ensembl_gene_id <- unlist(
  stri_split(
    str = Normalized_read_count$gene_id,
    regex = "\\\\.\\d*", # pattern is dot (.) followed by some number
    omit_empty = TRUE
  )
)
Normalized_read_count_newAnnotation <- merge(annotations,
Normalized_read_count,
                                           by = "ensembl_gene_id")

LIMK1_all_genes_newAnnotation <- merge(annotations, LIMK1_all_genes,
                                       by = "ensembl_gene_id")

annotations_protein_coding <- annotations[annotations$gene_biotype ==
"protein_coding", ]
LIMK1_all_protein_coding_genes <- merge(annotations_protein_coding,
LIMK1_all_genes,
                                       by = "ensembl_gene_id")

```

R code used for differential gene expression analysis of TCGA samples

```

## Data downloaded from
https://xenabrowser.net/datapages/?cohort=GDC%20TCGA%20Acute%20Myeloid%20Le
ukemia%20(LAML)&removeHub=https%3A%2F%2Fxcena.treehouse.gi.ucsc.edu%3A443,
March 08th, 2019
# Import expression (counts) data

TCGA.expression.AML <- read.table(file = "TCGA-LAML.htseq_counts.tsv.gz",
stringsAsFactors = FALSE, sep = "\t", header = TRUE, check.names = FALSE)
rownames(TCGA.expression.AML) <- TCGA.expression.AML[,1]
TCGA.expression.AML <- TCGA.expression.AML[, -1]

# Convert data to original counts (data are in log2(counts+1))
TCGA.expression.AML.counts <- 2^TCGA.expression.AML-1

coldata <- read.csv("coldata_LIMK1_TCGA.csv", header=TRUE, sep=";",
row.names = 1)
coldata$LIMK1 <- as.factor(coldata$LIMK1)
coldata$Sample <- rownames(coldata)
coldata <- coldata[order(coldata$Sample),]

# Samples in coldata only contain samples for which survival data are
available and those that represent the top 10% and bottom 10% in terms of
LIMK1 expression
# --> exclude the others from dataset to be consistent

samples <- as.data.frame(rownames(coldata))
colnames(samples) <- "Sample"

```

```

TCGA.expression.AML.counts26 <-
as.data.frame(t(TCGA.expression.AML.counts))
TCGA.expression.AML.counts26$Sample <-
rownames(TCGA.expression.AML.counts26)
TCGA.expression.AML.counts26_final <- merge(TCGA.expression.AML.counts26,
samples, by = "Sample", all = FALSE)
rownames(TCGA.expression.AML.counts26_final) <-
TCGA.expression.AML.counts26_final$Sample
TCGA.expression.AML.counts26_final <- TCGA.expression.AML.counts26_final[, -
1]
TCGA.expression.AML.counts26_final <-
as.data.frame(t(TCGA.expression.AML.counts26_final))
TCGA.expression.AML.counts26_final <-
round(TCGA.expression.AML.counts26_final, digits = 0)
TCGA.expression.AML.counts26_final$Ensembl_ID <-
rownames(TCGA.expression.AML.counts26_final)

# Pre-filtering based on TPM -> exclude genes with a mean TPM < 0.5
Input_TPM <- read.table(file = "TCGA-LAML.htseq_tpm.tsv.gz",
stringsAsFactors = FALSE, sep = "\t", header = TRUE, check.names = FALSE)
rownames(Input_TPM) <- Input_TPM$Ensembl_ID
Input_TPM <- Input_TPM[, -1]
Input_TPM$meanTPM <- rowMeans(Input_TPM)
TCGA_TPM_larger_0.5 <- Input_TPM[Input_TPM$meanTPM > 0.5, ]
TCGA_TPM_larger_0.5$Ensembl_ID <- rownames(TCGA_TPM_larger_0.5)
TCGA_TPM_larger_0.5_IDs <- as.data.frame(TCGA_TPM_larger_0.5$Ensembl_ID)
colnames(TCGA_TPM_larger_0.5_IDs) <- "Ensembl_ID"

TCGA.expression.AML.counts26_final_TPMfiltered <-
merge(TCGA.expression.AML.counts26_final, TCGA_TPM_larger_0.5_IDs, by =
"Ensembl_ID")
rownames(TCGA.expression.AML.counts26_final_TPMfiltered) <-
TCGA.expression.AML.counts26_final_TPMfiltered$Ensembl_ID
TCGA.expression.AML.counts26_final_TPMfiltered <-
TCGA.expression.AML.counts26_final_TPMfiltered[, -1]

# Check if coldata and expression data are in the same order
all(rownames(coldata) ==
colnames(TCGA.expression.AML.counts26_final_TPMfiltered))

# Make dds object
library("DESeq2")

dds.LIMK1.TCGA <- DESeqDataSetFromMatrix(countData =
TCGA.expression.AML.counts26_final_TPMfiltered,
colData = coldata,
design = ~ LIMK1)

dds.LIMK1.TCGA

rld <- vst(dds.LIMK1.TCGA)
plotPCA(rld, intgroup = "LIMK1",
returnData = FALSE)

# set reference level
dds.LIMK1.TCGA$LIMK1 <- relevel(dds.LIMK1.TCGA$LIMK1, ref = "low")

#Run analysis
dds.LIMK1.TCGA <- DESeq(dds.LIMK1.TCGA)

# QC
plotDispEsts(dds.LIMK1.TCGA)

```

```

# call results
resultsNames(dds.LIMK1.TCGA) # "Intercept" "LIMK1_high_vs_low"

res.TCGA.LIMK1_0.01 <- results(dds.LIMK1.TCGA, alpha = 0.01, name =
"LIMK1_high_vs_low")
summary(res.TCGA.LIMK1_0.01)

plotMA(res.TCGA.LIMK1_0.01)

# some low counts -> use IHW package for filtering
library(IHW)
res.TCGA.LIMK1_0.01_IHW <- results(dds.LIMK1.TCGA, name=
"LIMK1_high_vs_low", filterFun = IHW::ihw, alpha = 0.01)
summary(res.TCGA.LIMK1_0.01_IHW)
plotMA(res.TCGA.LIMK1_0.01_IHW)
res.TCGA.LIMK1_0.01_IHW <-
res.TCGA.LIMK1_0.01_IHW[order(res.TCGA.LIMK1_0.01_IHW$padj),]
res.TCGA.LIMK1_0.01_IHW$gene_id <- rownames(res.TCGA.LIMK1_0.01_IHW)
res.TCGA.LIMK1_0.01_IHW <- as.data.frame(res.TCGA.LIMK1_0.01_IHW)

## Annotate with gene symbols
# Remove version info from Ensembl gene IDs
library(stringi)
gene_ids_all <- unlist(
  stri_split(
    str = rownames(dds.LIMK1.TCGA),
    regex = "\\\\.\\d*", # pattern is dot (.) followed by some number
    omit_empty = TRUE
  )
)

head(gene_ids_all) # new variable which hasn't been assigned yet to a
matrix

# Map Ensembl gene IDs to gene names
library(biomaRt)
mart <- useEnsembl(
  biomart = "ensembl",
  dataset = "hsapiens_gene_ensembl",
  verbose = TRUE,
  mirror = "uswest"
)

mart

head(listAttributes(mart))

annotations <- getBM(
  attributes = c("ensembl_gene_id", "external_gene_name", "gene_biotype"),
  filters = "ensembl_gene_id",
  values = gene_ids_all,
  mart = mart
)

# Apply LFC shrinkage
results.LIMK1_TCGA_IHW_LFC <- lfcShrink(dds.LIMK1.TCGA, res =
res.TCGA.LIMK1_0.01_IHW,
                                coef = "LIMK1_high_vs_low",
                                type="apeglm")
summary(results.LIMK1_TCGA_IHW_LFC)

```

```

results.LIMK1_TCGA_IHW_LFC_df <- as.data.frame(results.LIMK1_TCGA_IHW_LFC)
results.LIMK1_TCGA_IHW_LFC_df$ensembl_gene_id <- unlist(
  stri_split(
    str = rownames(results.LIMK1_TCGA_IHW_LFC_df),
    regex = "\\\\.\\d*", # pattern is dot (.) followed by some number
    omit_empty = TRUE
  ))

results.LIMK1_TCGA_IHW_LFC_df_annotated <-
merge(results.LIMK1_TCGA_IHW_LFC_df, annotations, by = "ensembl_gene_id")
results.LIMK1_TCGA_IHW_LFC_df_annotated <-
results.LIMK1_TCGA_IHW_LFC_df_annotated[,c(1,7:8,2:6)]

# Filter for protein coding genes
results.LIMK1_TCGA_IHW_LFC_ProtCoding <-
results.LIMK1_TCGA_IHW_LFC_df_annotated[results.LIMK1_TCGA_IHW_LFC_df_annotated$gene_biotype=="protein_coding",]
write.csv(results.LIMK1_TCGA_IHW_LFC_ProtCoding, file =
"results.LIMK1_TCGA_IHW_LFC_protein_coding.csv", row.names = FALSE)

# Retrieve all genes with padj < 0.01 and log fold change < -1 or > 1
results.LIMK1_TCGA_IHW_LFC_sig <-
results.LIMK1_TCGA_IHW_LFC_df[results.LIMK1_TCGA_IHW_LFC_df$log2FoldChange
< -1 & results.LIMK1_TCGA_IHW_LFC_df$padj < 0.01 &

results.LIMK1_TCGA_IHW_LFC_df$baseMean > 10 |

results.LIMK1_TCGA_IHW_LFC_df$log2FoldChange > 1 &
results.LIMK1_TCGA_IHW_LFC_df$padj < 0.01 &

results.LIMK1_TCGA_IHW_LFC_df$baseMean > 10,]
results.LIMK1_TCGA_IHW_LFC_sig$ensembl_gene_id<- unlist(
  stri_split(
    str = rownames(results.LIMK1_TCGA_IHW_LFC_sig),
    regex = "\\\\.\\d*", # pattern is dot (.) followed by some number
    omit_empty = TRUE
  ))
results.LIMK1_TCGA_IHW_LFC_sig_annotated <-
merge(results.LIMK1_TCGA_IHW_LFC_sig, annotations, by = "ensembl_gene_id")
results.LIMK1_TCGA_IHW_LFC_sig_annotated <-
results.LIMK1_TCGA_IHW_LFC_sig_annotated[,c(1,7:8,2:6)]

# Export normalized read counts
dds.LIMK1.TCGA <- (estimateSizeFactors(dds.LIMK1.TCGA))
Normalized_read_count <- as.data.frame(counts(dds.LIMK1.TCGA,
normalized=TRUE))
Normalized_read_count$ensembl_gene_id <- gene_ids_all
Normalized_read_count_annotated <- merge(Normalized_read_count,
annotations, by = "ensembl_gene_id", all = TRUE)
Normalized_read_count_annotated <-
Normalized_read_count_annotated[,c(1,28:29,2:27)]

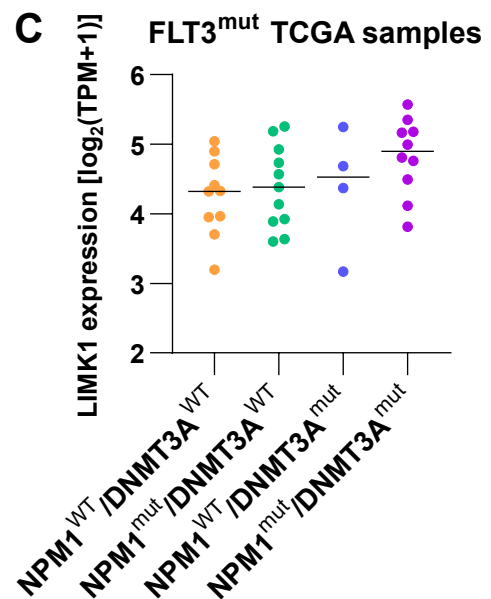
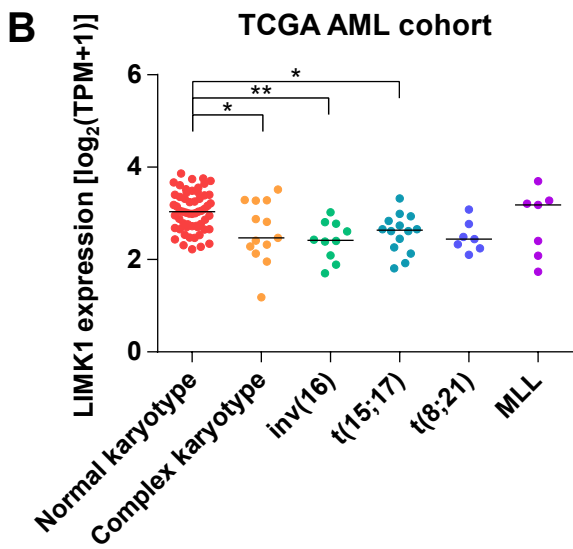
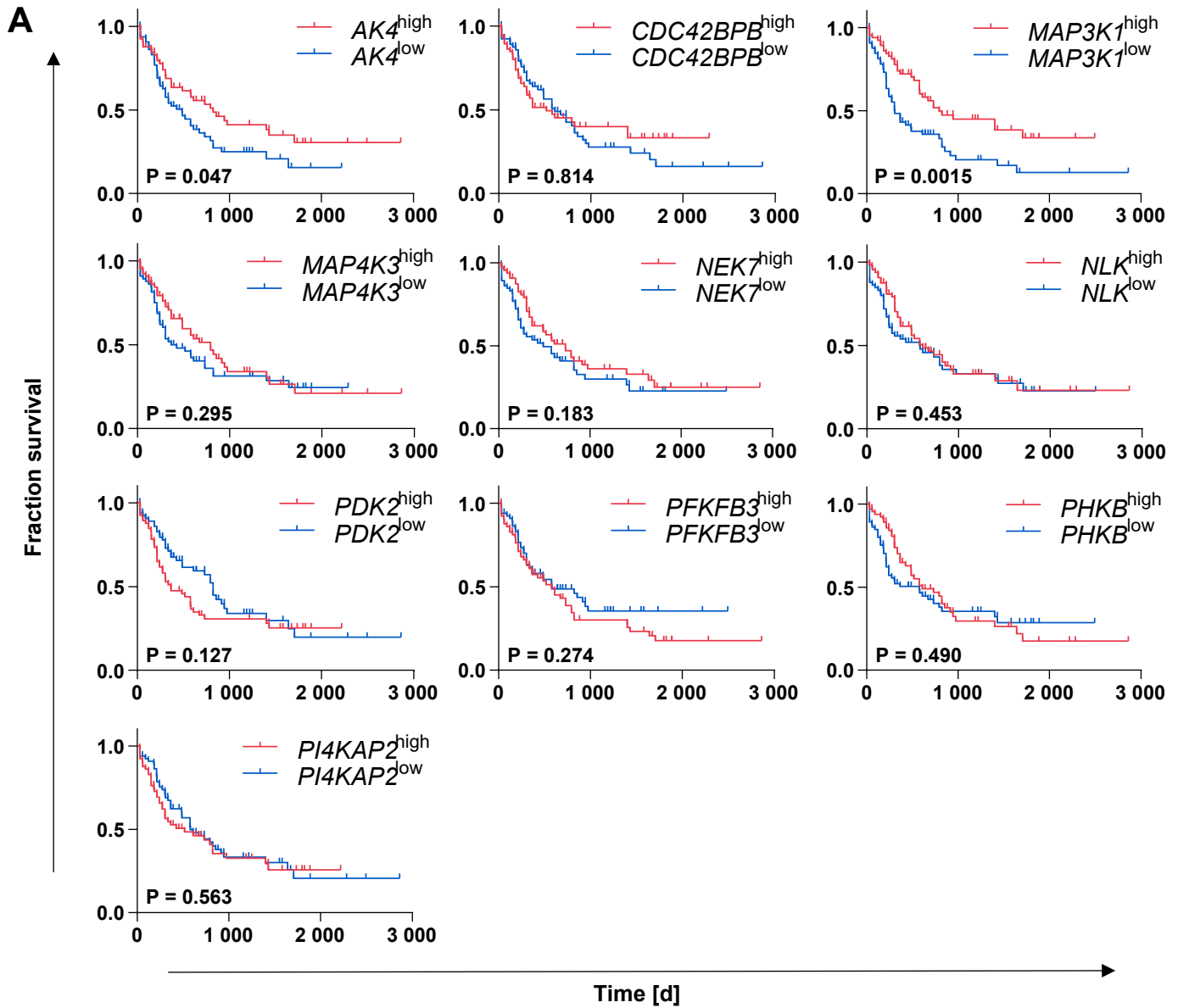
```

Supplemental references

1. Sanjana NE, Shalem O, Zhang F. Improved vectors and genome-wide libraries for CRISPR screening. *Nat Methods* 2014 Aug; **11**(8): 783-784.
2. Placke T, Faber K, Nonami A, Putwain SL, Salih HR, Heideel FH, *et al.* Requirement for CDK6 in MLL-rearranged acute myeloid leukemia. *Blood* 2014 Jul 3; **124**(1): 13-23.

3. Harrell Jr. FE. rms: Regression Modeling Strategies. R package version 5.1-4. 2019 [cited 2020 May 12]; Available from: <https://CRAN.R-project.org/package=rms>
4. Dobin A, Davis CA, Schlesinger F, Drenkow J, Zaleski C, Jha S, *et al.* STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* 2013 Jan 1; **29**(1): 15-21.
5. Liao Y, Smyth GK, Shi W. featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics* 2014 Apr 1; **30**(7): 923-930.
6. Liao Y, Smyth GK, Shi W. The Subread aligner: fast, accurate and scalable read mapping by seed-and-vote. *Nucleic Acids Res* 2013 May 1; **41**(10): e108.
7. Edgar R, Domrachev M, Lash AE. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res* 2002 Jan 1; **30**(1): 207-210.

Supplemental Figures

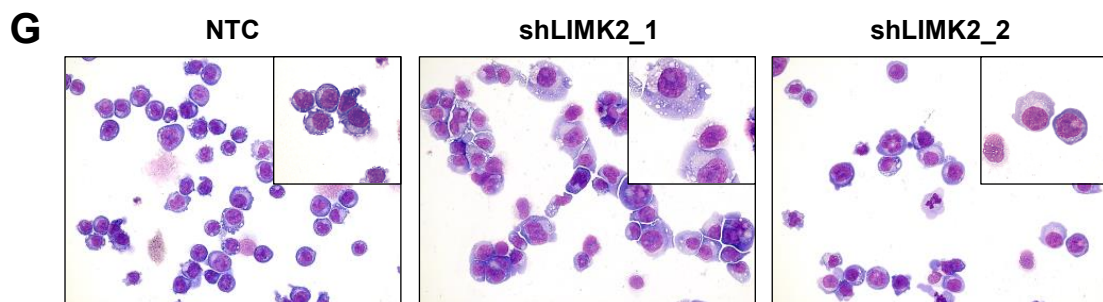
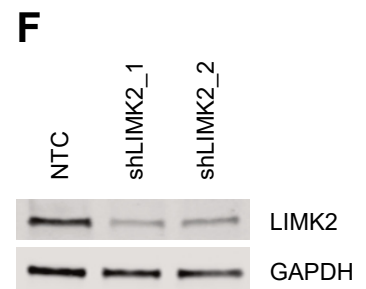
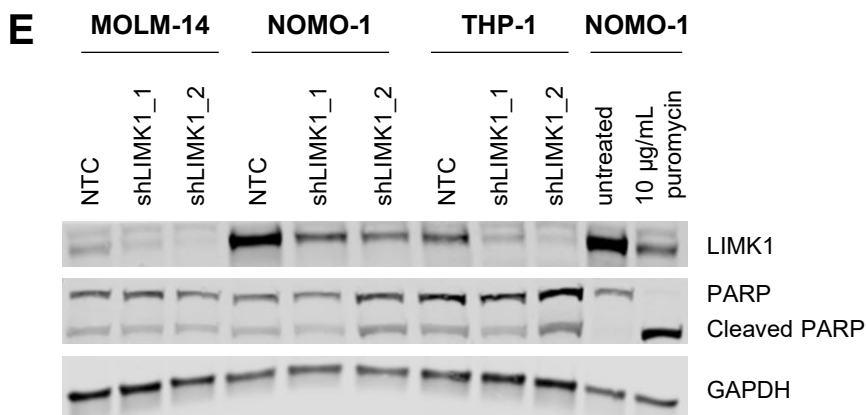
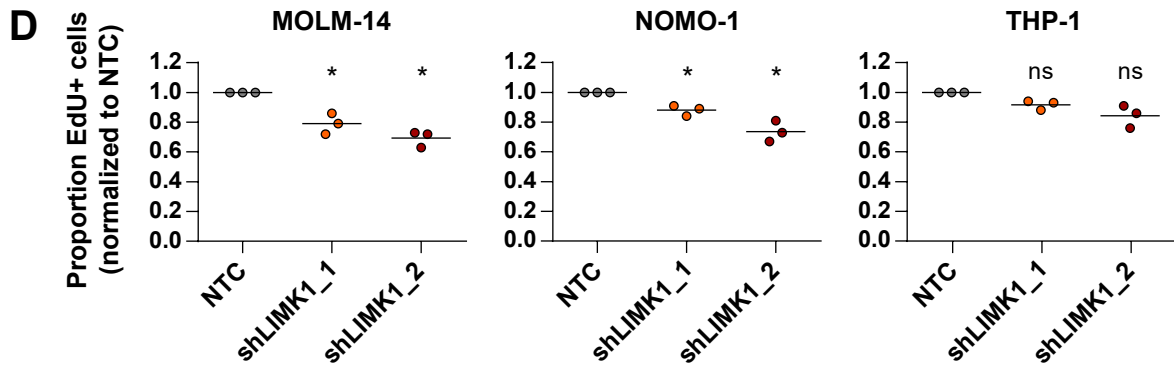
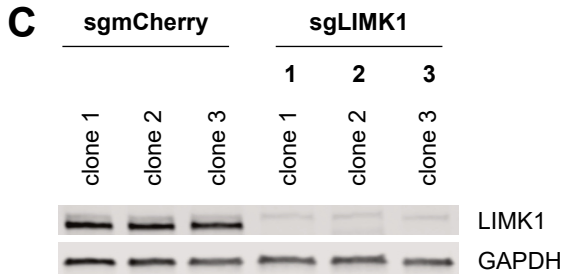
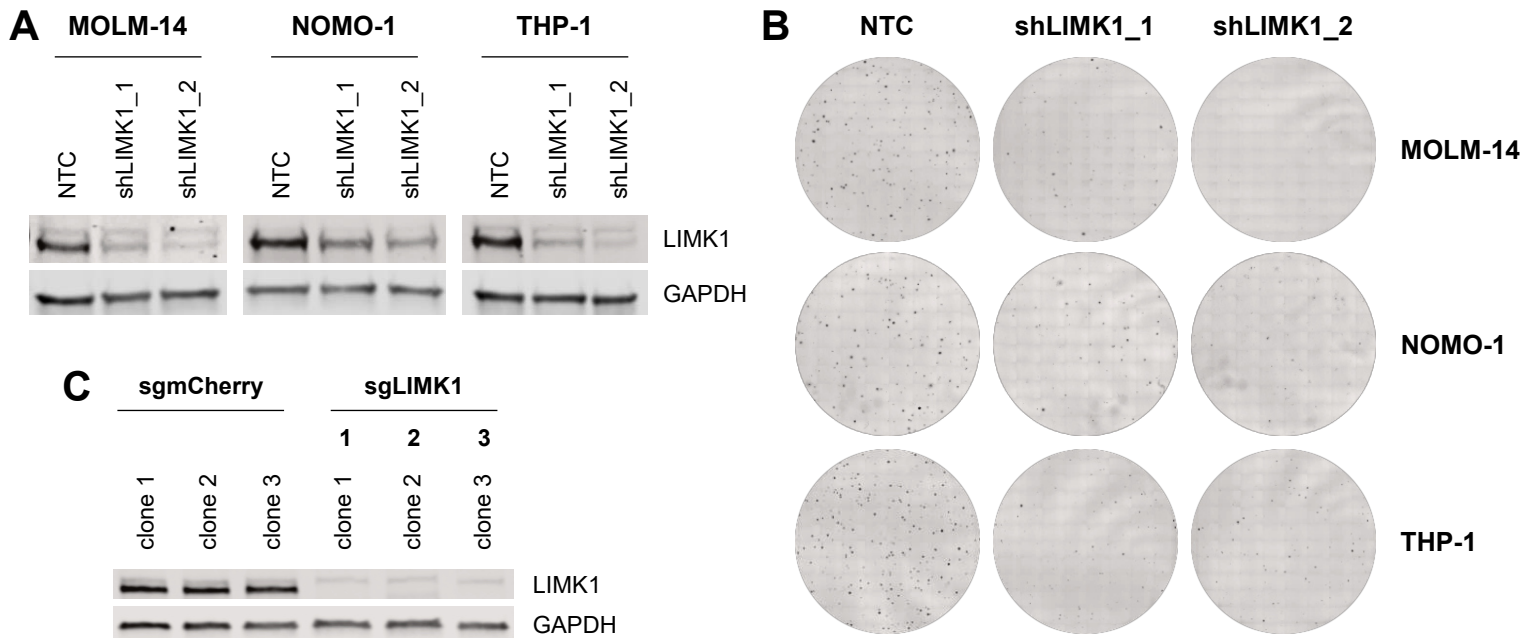


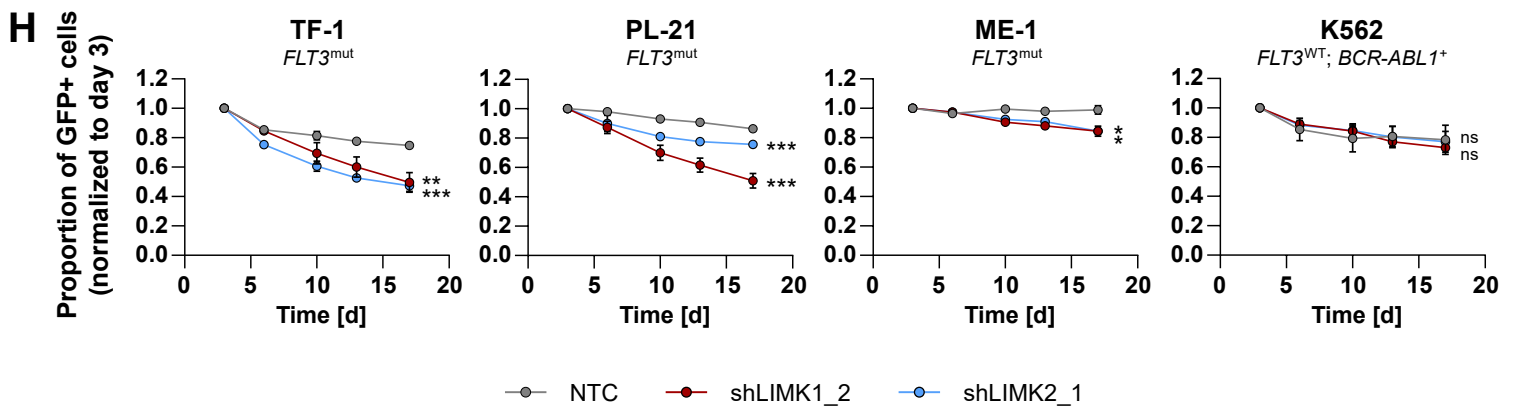
Supplemental Figure 1 (related to Figure 1).

(A) Survival of AML patients from the TCGA study (n=132) with candidate gene mRNA expression above the median and below the median. Statistical significance was assessed by log-rank test.

(B) *LIMK1* mRNA expression depending on cytogenetic subtype from the TCGA AML dataset (n=108; MLL = KMT2A). Black bars indicate the median. Statistical significance was assessed by one-way ANOVA with Tukey's correction for multiple comparisons (each genotype compared to all others). The non-indicated comparisons were not significant ($P > 0.05$). * $P < 0.05$, ** $P < 0.01$.

(C) *LIMK1* mRNA expression of *FLT3* mutant TCGA AML samples (n=35) depending on *NPM1* and *DNMT3A* mutation status. Black bars indicate the median. Statistical significance was assessed by Kruskal-Wallis test with Dunn's correction for multiple comparisons (each genotype compared to all others). Non-indicated comparisons were not significant ($P > 0.05$).





Supplemental Figure 2 (related to Figure 2).

(A) Western blots confirming efficient knockdown of LIMK1 in MOLM-14, NOMO-1, and THP-1 cells. Lysates were prepared twelve days post transduction.

(B) Representative images of the colony formation assays performed in MOLM-14, NOMO-1, and THP-1 cells (shown in Figure 2B).

(C) Western blot confirming LIMK1 knockout in THP-1-derived single cell clones.

(D) EdU incorporation assay in MOLM-14, NOMO-1, and THP-1 cells. Cells were incubated with EdU for eight hours. The assay was performed seven days after shRNA transduction. Data were normalized to NTC for each replicate. Statistical significance was assessed on log-transformed data by one-sample t-tests. * $P < 0.05$. ns, not significant ($P > 0.05$).

(E) Western blot for PARP and cleaved PARP upon LIMK1 knockdown in three *KMT2A*-rearranged AML cell lines. Lysates were prepared eleven days post transduction. NOMO-1 cells were treated with 10 $\mu\text{g}/\text{mL}$ puromycin for six hours as positive control for PARP cleavage.

(F) Western blot confirming efficient shRNA-mediated LIMK2 depletion in THP-1 cells. Lysates were prepared eleven days post transduction.

(G) May-Grünwald-Giemsa-stained cytopspin preparations of THP-1 cells eight days after transduction with two shRNAs targeting *LIMK2* or NTC. Original magnification, x 40. Insets show 1.5-fold magnified details of the corresponding photographs.

(H) Competition assays of three *KMT2A* wildtype, *FLT3* mutant AML cell lines and K562 cells (*KMT2A* wildtype, *FLT3* wildtype, *BCR-ABL1*⁺) transduced with two *LIMK1*-targeting shRNAs or NTC co-expressing GFP. GFP-positive cells were monitored by flow cytometry for 17 days starting three days post transduction. The experiment was performed three (TF-1, PL-21, K562) or two (ME-1) times, respectively. Statistical analysis was performed on log-transformed data for the final day of the experiment by unpaired t-tests. * $P < 0.05$, ** $P < 0.01$, *** $P < 0.001$. ns, not significant ($P > 0.05$).

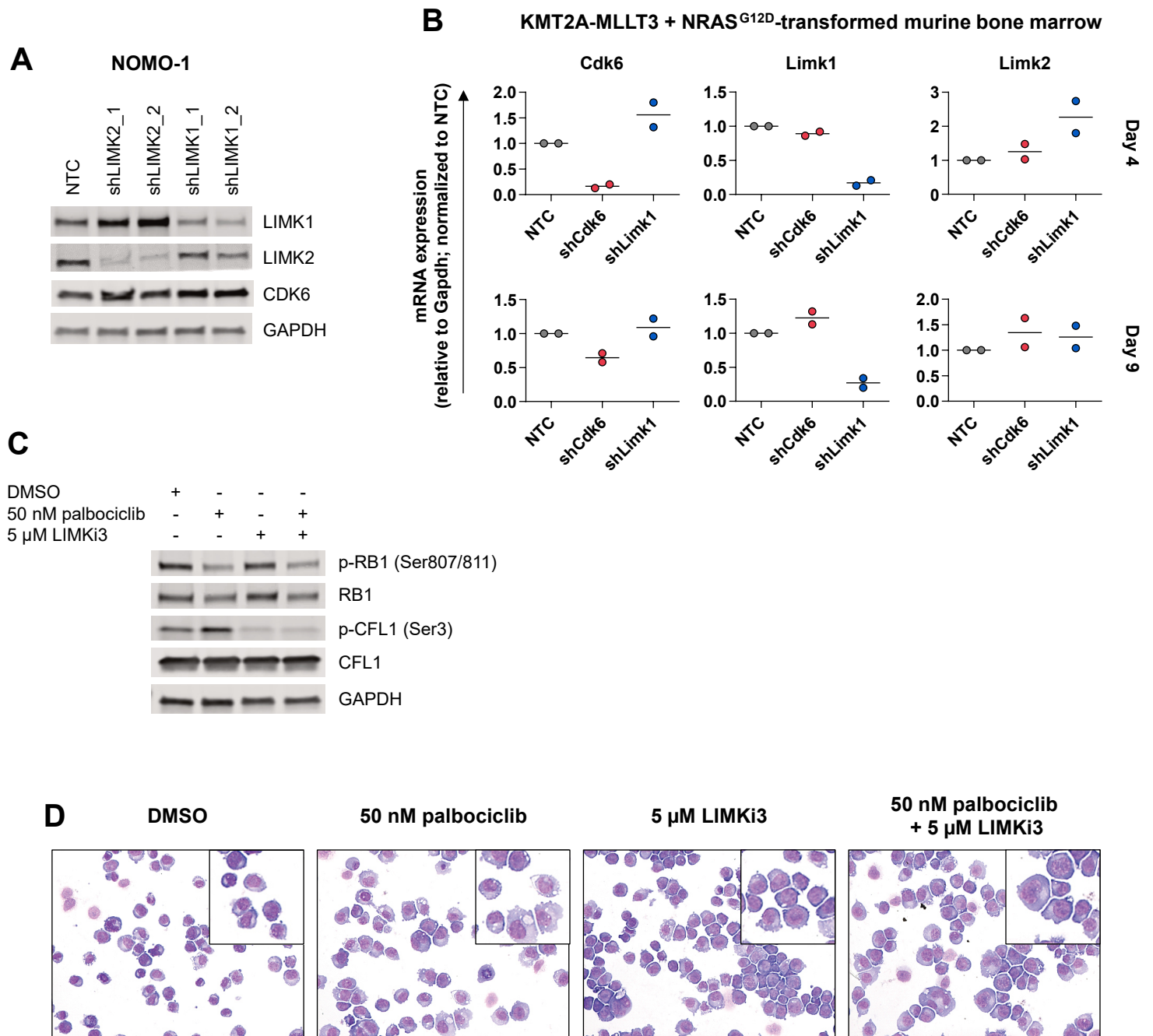
Supplemental Figure 3 (related to Figure 3).

(A) Depletion of cell cycle- and mitosis-related gene sets in LIMK1-depleted THP-1 cells. Genes were pre-ranked based on the mean $\log_2(\text{normalized read count shLIMK1}/\text{normalized read count NTC})$ of both shRNAs. KD, knockdown.

(B) Heatmap of *HOXA* and *HOXB* expression in AML patient samples from the TCGA study (*LIMK1*^{high}, n=13; *LIMK1*^{low}, n=13). TPM, transcripts per million.

(C) Correlation of *LIMK1* and *HOXA9* mRNA expression in AML patient samples from the TCGA study (n=132). Spearman's correlation coefficient is given. Statistical significance was assessed using the AS 89 algorithm. The semi-transparent area shows the 95% confidence interval. FPKM, fragments per kilobase million.

(D) RNA-seq data for *HOXA9* and its cofactors following LIMK1 suppression in THP-1 cells. TPM, transcripts per million.



Supplemental Figure 4 (related to Figure 4).

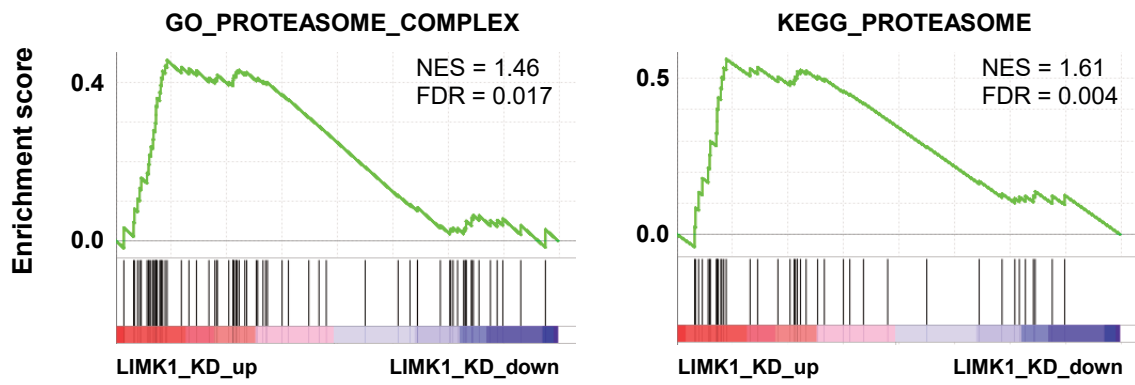
(A) Western blot of NOMO-1 cells transduced with LIMK1- or LIMK2-targeting shRNAs or NTC. Lysates were prepared eight days after transduction.

(B) qRT-PCR of murine bone marrow cells transformed with KMT2A-MLLT3 and NRAS^{G12D} (n=2).

Transduced cells were sorted two days after transduction. RNA was isolated four days (liquid culture, upper panel) or nine days (four days liquid culture + five days methylcellulose, lower panel) after transduction, respectively.

(C) Western blot confirming palbociclib and LIMKi3 efficacy after four days of treatment.

(D) May-Grünwald-Giemsa-stained cytopsin preparations of THP-1 cells treated with palbociclib, LIMKi3, or a combination of both for four days. Original magnification, x 40. Insets show 1.5-fold magnified details of the corresponding photographs.



Supplemental Figure 5.

Upregulation of proteasome-associated genes in LIMK1-depleted THP-1 cells. Genes were pre-ranked based on the mean $\log_2(\text{normalized read count shLIMK1}/\text{normalized read count NTC})$ of both shRNAs. KD, knockdown.