High-Throughput 5' UTR Engineering for Enhanced Protein Production in Non-Viral Gene Therapies

Jicong Cao[1, 2, 3, 4, 7], Eva Maria Novoa[4, 5, 6, 7 #], Zhizhuo Zhang[4, 5, 6], William C.W. Chen[1, 2, 3], Dianbo Liu[1, 4, 5], Gigi C G Choi[1, 2, 3 *], Alan S L Wong[1, 2, 3 *], Claudia Wehrspaun[1, 2, 3], Manolis Kellis[4, 5, 6 †], Timothy K Lu[1, 2, 3 ,4, 6 †]

[1]*Synthetic Biology Group, Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*

[2]*Department of Biological Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*

[3]*Synthetic Biology Center, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*

[4]*Broad Institute of MIT and Harvard, Cambridge, MA 02142, USA.*

[5]*Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*

[6]*Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*

[7]*The authors contributed equally to this work.*

[#]*Present address: Center for Genomic Regulation (CRG), 08003 Barcelona, Spain.*

[*]*Present address: School of Biomedical Sciences, University of Hong Kong, Hong Kong, China*

[†]*Corresponding authors: manoli@mit.edu, timlu@mit.edu*

ABSTRACT

Despite significant clinical progress in cell and gene therapies, maximizing protein expression in order to enhance potency remains a major challenge. One approach to increase protein expression is by optimizing translation through the engineering of 5' untranslated regions (5' UTRs). Here, we developed a high-throughput strategy to design, screen, and optimize novel 5'UTRs that enhance protein expression from a strong human cytomegalovirus (CMV) promoter. We first identified naturally occurring 5' UTRs with high translation efficiencies and used this information with *in silico* genetic algorithms to generate synthetic 5' UTRs. A total of ~12,000 5' UTRs were then screened using a recombinase-mediated integration strategy that greatly enhances the sensitivity of high-throughput screens by eliminating copy number and position effects that limit lentiviral approaches. Using this approach, we identified three synthetic 5' UTRs that outperformed commonly used non-viral gene therapy plasmids in expressing protein payloads. Furthermore, combinatorial assembly of these 5' UTRs enabled even higher protein expression than obtained with each individual 5' UTR. In summary, we demonstrate that high-throughput screening of 5' UTR libraries with recombinase-mediated integration can identify genetic elements that enhance protein expression, which should have numerous applications for engineered cell and gene therapies.

INTRODUCTION

In recent years, gene therapies that enable the exogenous production of proteins to replace defective genes have started to have transformative clinical impact[1–3]. Gene therapies can be delivered into patients through viral vectors or non-viral vectors. One of major challenges facing current gene therapy approaches is maximizing potency, since increasing the amount of exogenously expressed protein can reduce dose requirements and thus manufacturing costs, while improving human clinical results[4]. Multiple strategies are being employed to improve potency, such as enhancing cellular transduction efficiency by using more efficient viral vectors or non-viral transduction reagents, or by improving the gene expression construct itself. For example, recombinant adeno-associated virus (rAAV) is one of the major modalities used in gene therapy due to its infectivity and ability to achieve long-term gene expression *in vivo*[5,6]. However, 10^{16}-10^{17} genome copies (GCs) of rAAVs are being used in clinical trials, which requires the use of 100-10,000-L scale bioreactors to produce exceptionally large amounts of cGMP-grade viruses and is expensive[4,7]. Furthermore, a recent non-human primate study showed that high-dose intravenous rAAVs administration can lead to severe liver and neuronal toxicity[8]. Thus, there is a significant unmet need to improve protein production from viral gene therapies to improve the therapeutic window and reduce costs.

In addition, enhancing gene expression for non-viral DNA therapy remains a significant challenge. Non-viral gene therapy delivers DNA into cells to produce therapeutic proteins or vaccine antigens *in vivo*, with several potential advantages over viral gene therapies[9–11]. First, non-viral DNA therapy is potentially less immunogenic than viral particles since it uses chemical delivery strategies[12–14]. Second, the ability to deliver large amounts of DNA cargo via non-viral routes is greater than with viral vectors, where packaging limits place significant restrictions. Third, plasmids are relatively inexpensive to produce at the research and industrial scale and are more stable compared to viruses[1516]. The efficiency of *in vivo* DNA delivery has improved significantly as a result of recent advancements in liposome chemistry and nanoparticles, but is still not as efficient as viruses in many cases[17–19]. Thus, repeat dosing or increased dose levels have been attempted but these strategies can incur greater costs, risk of side effects, and sacrifice patient convenience.

In addition to optimizing delivery efficiencies, protein expression from gene therapies can be enhanced by optimizing the nucleic acid payload being delivered. Gene expression cassettes consist of multiple elements: promoter (which may include an enhancer), 5' untranslated region (5' UTR), protein coding region, 3' UTR, and polyadenylation (PolyA) signal[20]. Previous work has involved promoter engineering to enhance transcription or to enable cell-type specific gene expression[21,22]. However, fewer efforts to modulate translation through UTR engineering have been described.

Here, we focused on optimizing the 5'UTR to improve protein production in a non-viral gene therapy context. The rational design of 5' UTRs to enhance protein expression remains challenging, even though

regulatory elements and 5' UTR sequences that regulate gene expression in certain scenarios have been identified [23–27]. The design of 5' UTRs has been held back by limited knowledge of the relationships between 5' UTR sequences and associated levels of protein expression. In this study, we identified naturally occurring 5' UTRs with different translational activities in multiple human cell types. We then applied a genetic algorithm to obtain novel synthetic 5' UTRs, which were generated by evolving strong endogenous human 5' UTRs *in silico* (**Fig. 1**). To enable high-throughput testing of 12,000 distinct 5' UTRs, we developed a recombinase-based library screening strategy to eliminate copy number artefacts and positional effects, which introduce significant noise in traditional lentiviral-based library screening approaches[28,29]. Ultimately, we identified three synthetic 5' UTRs that significantly outperformed both naturally occurring 5' UTRs, as well as a commonly used non-viral gene therapy plasmid (pVAX1)[30]. Finally, we showed that the three synthetic 5' UTRs enhance protein expression across a variety of cell types and can be combined together to achieve further improvements, thus highlighting the potential of this approach for gene therapy applications.

RESULTS

5'UTR model training and library design by genetic algorithms

Protein production comprises two major steps: in the first step, DNA is transcribed into mRNA; and in the second step, mRNA is translated into protein. While transcription and translation are coupled in prokaryotic cells, these two steps are uncoupled in eukaryotic cells. Consequently, eukaryotic protein expression levels are highly dependent on mRNA levels, which are governed by the transcription machinery, but also on the translation efficiency (TE) of the transcripts, which is governed by the translation machinery[31,32]. In this context, given an identical transcription rate for two transcripts, the differences in the final amount of protein can be modulated by features found in the 5' UTR regions, which are involved in the recruitment of ribosomes[33]. The TE of a gene, i.e., the rate of mRNA translation into protein, can be calculated as the ratio of the ribosomal footprints (RPF) observed on a given mRNA of interest, which can be measured using Ribo-seq[34], to the relative abundance of that mRNA transcript in the cell, which can be measured using RNA-seq.

We first investigated the TEs of naturally occurring 5' UTRs (**Fig. 2**). To this end, we gathered publicly available Ribo-seq and RNA-seq data from human muscle tissue[35], as well as from two human cell lines, human embryonic kidney (HEK) 293T[36] and human prostate cancer cell line PC3[37]. A 5' UTR length of 100 bp was chosen and fixed for training algorithms and engineering 5' UTRs, which is compatible with the limits of current commercially available ssDNA template biosynthesis. For 5' UTR sequences that were longer than 100 bp, sequences were extracted from the 5' end and 3' end to construct two new 100-bp long

5'UTRs; those shorter than 100 bp were filled up with repeats of a CAA motif that does not have known secondary structure[38] to create two sequence versions, one having a shift of one nucleotide relative to the other (see Methods). AUGs were removed by randomly mutating one of the three nucleotides to avoid generating undesired upstream open reading frames (**Supplementary Table 1**).

Next, we computationally generated synthetic sequences by mutating and evolving endogenous 5' UTRs *in silico*. We trained and developed a computational model to predict TE based on 5' UTR characteristics (**Fig. 2**). Specifically, we extracted sequence features of 5' UTR regions that could be associated with gene expression levels and TE, which included k-mer frequency, codon usage, RNA folding energy, 5' UTR length, and number of ORFs. A random forest regression model was then trained on sequence features to predict TE and mRNA expression (**Supplementary Fig. 1**). The model was trained on experimentally determined TE rates and mRNA levels, which were obtained from analyzing publicly available RNA-seq and Ribo-seq data of endogenous genes from the three human cell types noted above: HEK 293T cells, PC3 cells, and human muscle tissue[39]. Given that searching for all 4^{100} possible 100-bp sequences would be too computationally demanding, we applied a genetic algorithm[40], which simulates the evolution process, to search for "optimal" sequences by mutating and recombining the endogenous sequences (see Methods). We created 2388 synthetic 5' UTRs that were predicted to have high TEs (**Supplementary Table 2**), in addition to a testing set designed by evolving 1198 5' UTRs with a range of TEs within 2 evolutionary generations (**Supplementary Table 3**). Overall, a total of 3586 synthetic sequences and 8414 naturally occurring sequences were used to build the ~12,000 100-bp 5'UTR library for this study.

Recombinase-mediated library screening to minimize copy number and position effects

Lentiviral-based library screening is the most commonly used method for high-throughput genetic screening[41–43]. In this method, diversified genetic elements are cloned into a lentiviral carrier plasmid and transfected into a virus-producing cell line with packaging and envelope plasmids to produce a lentiviral library, which is then used to infect the cells of interest. A multiplicity of infection (MOI) of ~0.1-0.3 is widely used to ensure that most of infected cells receive only one copy of the element of interest. However, even at 0.1 MOI, 10% of the cells receive two or more copies. Moreover, lentiviruses insert randomly into the cellular genome, resulting in significant variations in gene expression[44,45]. As a result, a significant amount of noise due to copy number variations in cells and positional effects can obscure accurate phenotypic assessment of genetic constructs in lentiviral screens.

To address this issue, we designed a recombinase-based gene integration strategy to screen the 5' UTR library; this strategy ensures single-copy integration within each cell at a defined "landing-pad" location (**Fig. 3A**). We used the serine recombinase Bxb1 to integrate a plasmid containing the Bxb1 attB site into

the Bxb1 attP site on the genome, which results in destruction of the attP site to prevent additional insertions[46–49].

We first constructed HEK 293T cell lines with a landing pad by lentiviral infection. In these cell lines, the landing pad comprised a constitutive promoter, a mutant BxbI attP site with enhanced integration efficiency[50], and a yellow fluorescent protein (YFP)[47] as a reporter for the integration of the landing pad. Nine cell clones with insertion of the landing pad were identified and expanded. We chose to use two different cell lines with different YFP expression levels during our screens to reduce the impact of genomic location on the screening phenotype. The 5'UTR library was cloned upstream of the GFP reporter on the payload plasmid, which also encoded a BxB1 attB site and a red fluorescent protein (RFP) and puromycin duo selection marker (**Fig. 3B**). In this system, successful integration activates expression of RFP and puromycin and inhibits YFP expression. We integrated the 5'UTR library into the two different landing pad cell lines with >25-fold more cells than the size of the library (>300k integrated cells). The transfected cells were grown for one week, then subjected to puromycin selection for another 4 days.

To identify 5'UTRs with increased protein expression, we used FACS to sort the cell library into four bins based on GFP expression levels: top 2.5%, 2.5-5%, 5-10%, and 0-100% (unsorted). We then extracted genomic DNAs from cells in each bin and optimized PCR conditions for unbiased amplicon amplification[41]. The amplicons were then barcoded and sequenced using Illumina NextSeq. We calculated the relative abundance of each 5' UTR sequence in each of the three top bins (2.5%, 2.5-5%, and 5-10%) and normalized them to the counts in the control bin (0-100%). Log2 ratios were used to represent the enrichment of each 5' UTR in each bin.

Our results showed that this recombinase-based library screening approach achieved Pearson correlation values greater than 0.93 between results obtained from the two landing pad cell lines in all three bins, thus demonstrating the high reproducibility of the screening process (**Fig. 3C**). This level of reproducibility exceeded that of traditional lentiviral-based library screening, which had correlation values equal to 0.49, 0.49 and 0.54 for each of the three bins, respectively. Overall, our results clearly show that recombinase-based integration can significantly improve the reproducibility of high-throughput screening.

Validation of the 5' UTR hits in HEK 293T cells

To select candidates for further experimental validation, we ranked the 5' UTRs based on their relative expression levels in top expressing bins (2.5%, 2.5-5%, and 5-10%), relative to the unsorted bin. Specifically, differentially enriched 5'UTRs in the top-expressing bins were determined using DESeq2[51], which takes into account variability across biological replicates to identify differentially expressed candidates. Top candidates were defined as 5' UTRs that showed at least 50% increased expression level in all three top-expressing bins (fold change greater than 50%) with significant adjusted p-values in all three

bins (p-adj<0.05) (**Supplementary Tables 4-6)**, relative to control (**Fig. 4A**). Using these criteria, thirteen 5' UTRs with enriched expression in all three bins were used for further validation. Interestingly, six of these thirteen 5' UTRs were synthetic 5'UTRs, implying a 300% enrichment towards synthetic sequences predicted to have high TEs (six out of 2388) compared to the pool of natural 5' UTRs included in the screening (seven out of 8414).

We then tested the selected 5'UTR candidates in the pVAX1 non-viral gene therapy plasmid[30]. pVAX1 has a human CMV promoter for high-level protein expression, a multiple cloning site for foreign gene insertion, and a bovine growth hormone (bGH) PolyA signal for transcriptional termination. We synthesized and inserted the thirteen candidate 5' UTRs (100 bp long) along with a green fluorescent protein (GFP) reporter (with Kozak sequence; pVAX1-UTR-GFP) downstream of the CMV promoter in the pVAX1 plasmid. As a control, we used only the GFP reporter (with Kozak sequence; pVAX1-GFP) (**Fig. 4B**). We co-transfected HEK 293T cells with the engineered 5'UTR-containing plasmids and the control plasmid along with a blue fluorescent protein (BFP) expression plasmid. This allowed us to normalize GFP expression to transfection efficiency, as determined by BFP levels. Six out of the thirteen tested plasmids showed higher GFP expression than the commercial protein expression plasmid pVAX1 in HEK 293T cells (**Supplementary Fig. 2**).

To demonstrate the potential therapeutic utility of these UTRs, we expressed two different therapeutic proteins with these 5' UTRs: vascular endothelial growth factor (VEGF), which stimulates the formation of blood vessels[58]; and C-C motif chemokine ligand 21 (CCL21), which can recruit immune cells for immunotherapy[59]. Two out of the three 5' UTRs increased VEGF expression compared to the commercial plasmid, and one of these, NeoUTR2, increased VEGF production by 42% relative to pVAX1 ($p = 0.01$) (**Fig. 4E**). All three 5' UTRs increased CCL21 expression by greater than 100% relative to pVAX1 ($p = 0.02$, 0.04, 0.0002, respectively), and NeoUTR3 showed an impressive increase of 452% (**Fig. 4F**). In summary, we identified three novel 5' UTR thru *in silico* design and high-throughput screening that significantly increase protein expression levels for fluorescent protein reporters (by up to 58%) and therapeutic proteins (by up to 452%) from non-viral gene therapy vectors.

Combinatorial synthetic 5' UTRs can further enhance protein expression

Considering our promising results using synthetic 5'UTRs, we sought to investigate whether novel combinations of these 5'UTRs might further enhance GFP expression levels (**Fig. 5A**). Using our three NeoUTR leads as building blocks, we constructed six combinatorial 5' UTRs (CoNeoUTRs) by joining two of the NuUTRs with a 6-nt linker (CAACAA). These were labeled as CoNeoUTR2-3 (NeoUTR2-NeoUTR3), CoNuUTR1-3 (NeoUTR1-NeoUTR3), CoNeoUTR3-2 (NeoUTR3-NeoUTR2), CoNeoUTR1-2 (NeoUTR1-NeoUTR2), CoNeoUTR3-1 (NeoUTR3-NeoUTR1), and CoNeoUTR2-1 (NeoUTR2-

NeoUTR1). We inserted each combinatorial 5' UTR upstream of the GFP coding sequence, co-transfected HEK 293T cells with the resulting plasmids and the BFP expression plasmid, and then measured GFP and BFP fluorescence (**Fig. 5B**). We observed that the strength of the 5' UTR combinations was positively correlated with the strengths of the two individual 5' UTRs (**Supplementary Fig. 4**) Moreover, we observed that for the CoNeoUTRs constructed with two different NeoUTRs, the strength was higher if the stronger NeoUTR was placed at the 3' end: CoNeoUTR1-2 > CoNeoUTR2-1, CoNeoUTR1-3 > CoNeoUTR3-1, and CoNeoUTR2-3 > CoNeoUTR3-2.

Finally, we tested how the artificial 5' UTR elements modulate gene expression in different cell types. We chose: *i)* human breast cancer cell line MCF-7, *ii)* human muscle cancer rhabdomyosarcoma (RD) cells and *iii)* mouse muscle cell line C2C12. We found that all three 100-bp artificial 5' UTRs (NeoUTR1, NeoUTR2 and NeoUTR3) enhanced protein expression in the three cell types and HEK 293T cells; however, the relative strengths of the 5' UTRs were different in different cell types (**Fig. 5C**). Overall, 78% of all conditions tested, the synthetic 5'UTRs were statistically stronger than pVAX1 (the p-values are labeled in Fig. 5C) across the four cell types. Thus, these results show that the novel 5' UTR sequences and their combinatorial counterparts identified in this study can significantly enhance protein expression across a variety of mammalian cell types, further validating the applicability of our approach.

In summary, synthetic 5'UTRs can enhance protein production across multiple cell types and can be combined together to further modulate protein levels.

DISCUSSION

In this study, we developed a robust strategy for the systematic discovery and engineering of 5' UTRs for enhanced protein expression. We trained a computational model using gene expression information on naturally occurring 5' UTRs and evolved a novel synthetic 5'UTR library. We developed a recombinase-based high-throughput screening platform to overcome significant heterogeneity that limits the accuracy of lentiviral-based screens. The serine recombinase BxbI integrates one copy of tagged genetic elements at a specific location in the host genome, eliminating the copy number and position effects that are seen in conventional lentiviral-based library screening. We observed high reproducibility of this recombinase-based library screening strategy for 5' UTR engineering, allowing us to identify three synthetic 5' UTR candidates that increase protein production across multiple cell types. This strategy allowed us to identify synthetic 5' UTRs that outperformed the commonly used pVAX1 vector and 4 commonly used introns in terms of their ability to increase protein production as non-viral gene delivery vectors.

Although the synthetic 5' UTRs we validated in this study were strong across multiple contexts, their relative performance did vary depending on cell type. To optimize gene therapy for specific types of cells,

it could be useful to repeat the strategy employed here but focused on the cells of interest. In addition, future work should test whether these synthetic 5' UTRs work in other contexts, including AAV and lentiviral vectors, and with additional payloads. Finally, optimized 5' UTRs need to be ultimately validated *in vivo* for clinical translation and may need to take into account translational efficiencies across multiple model systems beyond human cells to ensure translatability.

ACKNOWLEDGEMENTS

The authors thank Dr. Stuart Levine at MIT MicroBio Center for assisting with NGS, and Karen Pepper for help with paper editing. The work was financially supported by Army Research Office (funding under the OSP account 6924758), Boston University (funding under the OSP account 6924758), HFSP to TKL, and NIH (R01 GM113708, R01 HG004037) to MK. EMN thanks Human Frontier Science Program (LT000307/2013-L) for their financial support.

AUTHOR CONTRIBUTIONS

JC, GCGC, EMN, MK and TKL conceived idea and designed the study. EMN, ZZ, GCGC designed the 5' UTR library. JC designed and performed experiments and analyzed data. EMN performed the NGS data pre-processing. ZZ and DL developed the computational analysis. JC, EMN, ZZ and DL performed the computational analysis. WC performed the ELISA experiments. GCGC and ASLW helped with library cloning. JC, ZZ, GCGC, EMN, MK and TKL wrote the paper. All authors discussed the results and edited the manuscript.

COMPETING INTERESTS

JC, EMN, ZZ, MK and TKL have filed patent applications on the work. TKL is a co-founder of Senti Biosciences, Synlogic, Engine Biosciences, Tango Therapeutics, Corvium, BiomX, and Eligo Biosciences. TKL also holds financial interests in nest.bio, Ampliphi, and IndieBio. The other authors declare no competing interests.

REFERENCES

1. Dunbar, C. E. *et al.* Gene therapy comes of age. *Science* **359**, 175 (2018).
2. Sheridan, C. Gene therapy finds its niche. *Nat. Biotechnol.* **29**, 121-128 (2011).
3. Kumar, S. R., Markusic, D. M., Biswas, M., High, K. A. & Herzog, R. W. Clinical development of gene therapy: results and lessons from recent successes. *Mol. Ther. - Methods Clin. Dev.* **3**, 16034(2016)
4. Joshi, P. R. H. *et al.* Achieving High-Yield Production of Functional AAV5 Gene Delivery Vectors via Fedbatch in an Insect Cell-One Baculovirus System. *Mol. Ther. - Methods Clin. Dev.* **16**, 279-289 (2019).
5. Naso, M. F., Tomkowicz, B., Perry, W. L. & Strohl, W. R. Adeno-Associated Virus (AAV) as a Vector for Gene Therapy. *BioDrugs* **31**, 317-334 (2017).
6. Waehler, R., Russell, S. J. & Curiel, D. T. Engineering targeted viral vectors for gene therapy. *Nat. Rev. Genet.* **8**, 573-587 (2007).
7. George, L. A. Hemophilia gene therapy comes of age. *Hematology Am. Soc. Hematol. Educ. Program.* **2017**, 587-594 (2017).
8. Hinderer, C. *et al.* Severe Toxicity in Nonhuman Primates and Piglets Following High-Dose Intravenous Administration of an Adeno-Associated Virus Vector Expressing Human SMN. *Hum. Gene Ther.* **29**, 285-298 (2018).
9. Ramamoorth, M. & Narvekar, A. Non viral vectors in gene therapy - An overview. *J. Clin. Diagn. Res.* **9**, GE01-06 (2015).
10. Yin, H. *et al.* Non-viral vectors for gene-based therapy. *Nat. Rev. Genet.* **15**, 541-555(2014).
11. Hardee, C. L., Arévalo-Soliz, L. M., Hornstein, B. D. & Zechiedrich, L. Advances in non-viral DNA vectors for gene therapy. *Genes* **10**, 8 (2017).
12. Hacein-Bey-Abina, S., Fischer, A. & Cavazzana-Calvo, M. Gene therapy of X-linked severe combined immunodeficiency. *Int. J. Hematol.* **24**, 580-584 (2002).
13. Hacein-Bey-Abina, S. *et al.* A Serious Adverse Event after Successful Gene Therapy for X-Linked Severe Combined Immunodeficiency. *N. Engl. J. Med.* ***384, 255-256*** (2003).
14. Escors, D. & Breckpot, K. Lentiviral vectors in gene therapy: Their current status and future potential. *Arch. Immunol. Ther. Exp.* **58**, 107-119 (2010).
15. Schmeer, M., Buchholz, T. & Schleef, M. Plasmid DNA Manufacturing for Indirect and Direct Clinical Applications. *Hum. Gene Ther.* **28**, 856-861 (2017).
16. Rodrigues, G. A. *et al.* Pharmaceutical Development of AAV-Based Gene Therapy Products for the Eye. *Pharm. Res.* **36**, 29 (2019).

17. Jones, C. H., Hill, A., Chen, M. & Pfeifer, B. A. Contemporary Approaches for Nonviral Gene Therapy. *Discov. Med.* (2015).
18. Shim, G. *et al.* Nonviral Delivery Systems For Cancer Gene Therapy: Strategies And Challenges. *Curr. Gene Ther.* **18**, 3-20(2018).
19. Bai, H., Lester, G. M. S., Petishnok, L. C. & Dean, D. A. Cytoplasmic transport and nuclear import of plasmid DNA. *Biosci. Rep.* **37**, 6 (2017).
20. Dronadula, N. *et al.* Construction of a novel expression cassette for increasing transgene expression in vivo in endothelial cells of large blood vessels. *Gene Ther.* **18**, 501-508(2011).
21. Wu, M. R. *et al.* A high-throughput screening and computation platform for identifying synthetic promoters with enhanced cell-state specificity (SPECS). *Nat. Commun.* **10**, 2880 (2019).
22. Ho, S. C. L. & Yang, Y. Identifying and engineering promoters for high level and sustainable therapeutic recombinant protein production in cultured mammalian cells. *Biotechnol. Lett.* **36**, 1569-1579 (2014).
23. Asrani, K. H. *et al.* Optimization of mRNA untranslated regions for improved expression of therapeutic mRNA. *RNA Biol.* **15**, 756-762(2018).
24. Weinberger, A. *et al.* Deciphering the rules by which 5'-UTR sequences affect protein expression in yeast. *Proc. Natl. Acad. Sci.* **110**, E2792-801 (2013).
25. Decoene, T., Peters, G., De Maeseneire, S. L. & De Mey, M. Toward Predictable 5′UTRs in Saccharomyces cerevisiae: Development of a yUTR Calculator. *ACS Synth. Biol.* **7**, 622-634 (2018).
26. Ding, W. *et al.* Engineering the 5′ UTR-Mediated Regulation of Protein Abundance in Yeast Using Nucleotide Sequence Activity Relationships. *ACS Synth. Biol.* **7**, 2709-2714 (2018).
27. Sample, P. J. *et al.* Human 5 ′ UTR design and variant effect prediction from a massively parallel translation assay. *Nat. Biotechnol.* **37**, 803-809 (2019).
28. Matreyek, K. A., Stephany, J. J. & Fowler, D. M. A platform for functional assessment of large variant libraries in mammalian cells. *Nucleic Acids Res.* **45**, e102 (2017).
29. Duportet, X. *et al.* A platform for rapid prototyping of synthetic gene networks in mammalian cells. *Clin. Cancer Res.* **24**, 6015-6027 (2018).
30. Muthumani, K. *et al.* Optimized and enhanced DNA plasmid vector based in vivo construction of a neutralizing anti-HIV-1 envelope glycoprotein Fab. *Hum. Vaccines Immunother.* **9**, 2253-2262 (2013).
31. Kozak, M. Downstream secondary structure facilitates recognition of initiator codons by eukaryotic ribosomes. *Proc. Natl. Acad. Sci.* **87**, 8301-8305 (1990).
32. Kozak, M. An analysis of 5'-noncoding sequences from 699 vertebrate messenger rNAS. *Nucleic Acids Res.* **15**, 8125-8148 (1987).

33. Ingolia, N. T. Ribosome profiling: New views of translation, from single codons to genome scale. *Nat. Rev. Genet.* **15**, 205-213 (2014).
34. Ingolia, N. T., Ghaemmaghami, S., Newman, J. R. S. & Weissman, J. S. Genome-wide analysis in vivo of translation with nucleotide resolution using ribosome profiling. *Science **324**, 218-223* (2009).
35. Ardlie, K. G. *et al.* The Genotype-Tissue Expression (GTEx) pilot analysis: Multitissue gene regulation in humans. *Science* **348**, 648-660 (2015).
36. Andreev, D. E. *et al.* Translation of 5' leaders is pervasive in genes resistant to eIF2 repression. *Elife* **4**, e03971 (2015).
37. Hsieh, A. C. *et al.* The translational landscape of mTOR signalling steers cancer initiation and metastasis. *Nature* **485**, 55-61 (2012).
38. Hanson, S., Berthelot, K., Fink, B., McCarthy, J. E. G. & Suess, B. Tetracycline-aptamer-mediated translational regulation in yeast. *Mol. Microbiol.* **49**, 1627-1637 (2003).
39. Ho, T. K. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.* **20**, 832-844 (1998).
40. Scrucca, L. GA : A Package for Genetic Algorithms in R. *J. Stat. Softw.* **53**, 1–37 (2015).
41. Wong, A. S. L., Choi, G. C. G., Cheng, A. A., Purcell, O. & Lu, T. K. Massively parallel high-order combinatorial genetics in human cells. *Nat. Biotechnol.* **33**, 952-961 (2015).
42. Chang, K., Elledge, S. J. & Hannon, G. J. Lessons from Nature: microRNA-based shRNA libraries. *Nat. Methods* **3**, 707-714. (2006).
43. Shalem, O. *et al.* Genome-scale CRISPR-Cas9 knockout screening in human cells. *Science* 343, 84-87 (2014).
44. Akhtar, W. *et al.* Chromatin position effects assayed by thousands of reporters integrated in parallel. *Cell* **154,** 914-27 (2013).
45. Wilson, C. Position Effects On Eukaryotic Gene Expression. *Annu. Rev. Cell Dev. Biol.* 6, 679-714 (1990).
46. Roquet, N., Soleimany, A. P., Ferris, A. C., Aaronson, S. & Lu, T. K. Synthetic recombinase-based State machines in living cells. *Science* **353**, aad8559 (2016).
47. Guye, P., Li, Y., Wroblewska, L., Duportet, X. & Weiss, R. Rapid, modular and reliable construction of complex mammalian gene circuits. *Nucleic Acids Res.* **41**, e156 (2013).
48. Perez-Pinera, P. *et al.* Synthetic biology and microbioreactor platforms for programmable production of biologics at the point-of-care. *Nat. Commun.* **7**, 12211 (2016).
49. Brown, W. R. A., Lee, N. C. O., Xu, Z. & Smith, M. C. M. Serine recombinases as tools for genome engineering. *Methods* **53**, 372-9 (2011).
50. Jusiak, B. *et al.* Comparison of Integrases Identifies Bxb1-GA Mutant as the Most Efficient Site-

Specific Integrase System in Mammalian Cells. *ACS Synth. Biol.* **8**, 16-24 (2019).

51. Love, M. I., Huber, W. & Anders, S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biol.* **15**, 550 (2014).
52. Lee, S. H., Danishmalik, S. N. & Sin, J. I. DNA vaccines, electroporation and their applications in cancer treatment. *Hum. Vaccines Immunother.* **11**, 1889-1900 (2015).
53. Kim, T. J. *et al.* Clearance of persistent HPV infection and cervical lesion by therapeutic DNA vaccine in CIN3 patients. *Nat. Commun.* **5**, 5317 (2014).
54. Https://bioscience.lonza.com/lonza_bs/US/en/Transfection/p/000000000000191671/pmaxCloning-Vector.
55. Kang, M. *et al.* Human β-globin second intron highly enhances expression of foreign genes from murine cytomegalovirus immediate-early promoter. *J. Microbiol. Biotechnol.* **15**, 544-550(2005).
56. Mariati, Ho, S. C. L., Yap, M. G. S. & Yang, Y. Evaluating post-transcriptional regulatory elements for enhancing transient gene expression levels in CHO K1 and HEK293 cells. *Protein Expr. Purif.* **69**, 9-15 (2010).
57. Xia, W. *et al.* High levels of protein expression using different mammalian CMV promoters in several cell lines. *Protein Expr. Purif.* **45**, 115-24 (2006).
58. Johnson, K. E. & Wilgus, T. A. Vascular Endothelial Growth Factor and Angiogenesis in the Regulation of Cutaneous Wound Repair. *Adv. Wound Care* **3**, 647-661 (2014).
59. Lin, Y., Sharma, S. & John, M. S. CCL21 cancer immunotherapy. *Cancers* **6**, 1098-1110 (2014).

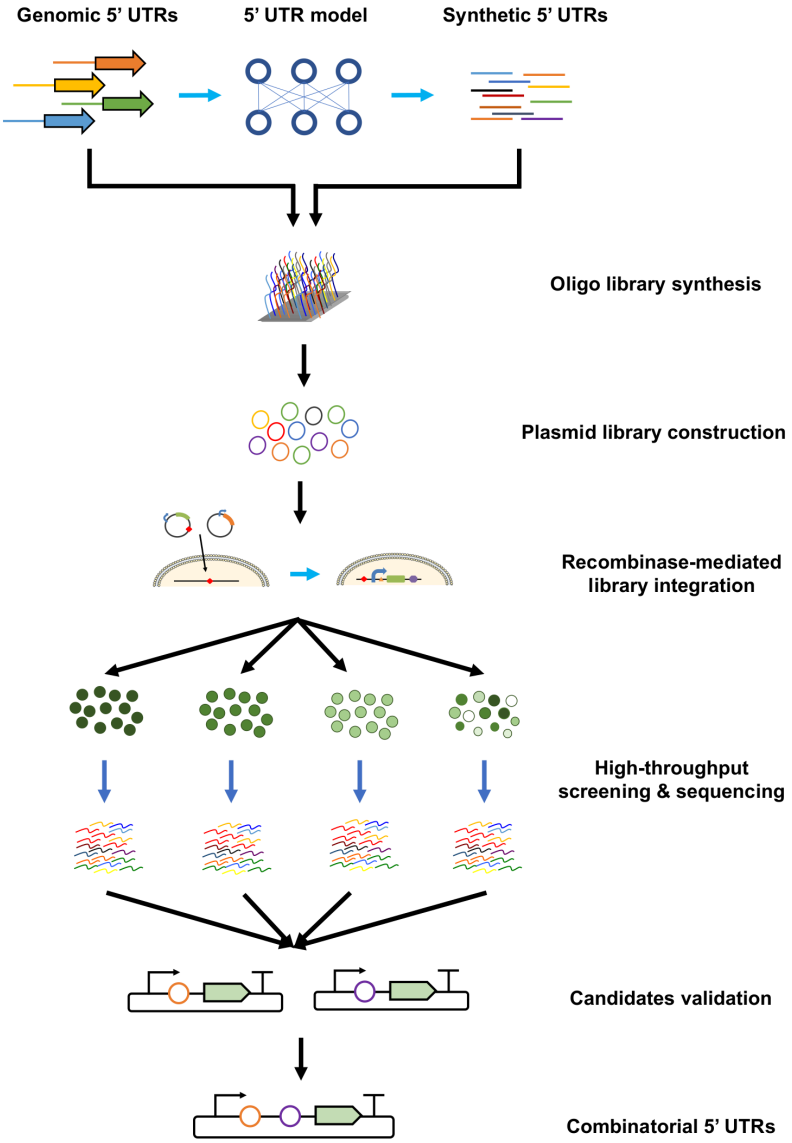


Figure 1. Schematic overview of recombinase-mediated 5' UTR library screening strategy. Naturally occurring 5' UTRs were extracted, analyzed, and used as the training set to generate synthetic 5' UTRs for screening. Oligos encoding the 5' UTR library were synthesized and cloned into plasmids containing a recombinase-recognition site and a GFP reporter. The resulting plasmids were transfected into the HEK 293T-LP cell line with the corresponding recombinase recognition site, resulting in targeted genomic insertion. The cells were sorted into bins based on GFP intensities, and the 5' UTR sequences of each bin were amplified, sequenced, counted, and compared. The 5' UTR candidates that enhanced GFP expression were selected and validated experimentally. Finally, the top-ranked validated 5' UTRs were combined to test for increased gene expression.

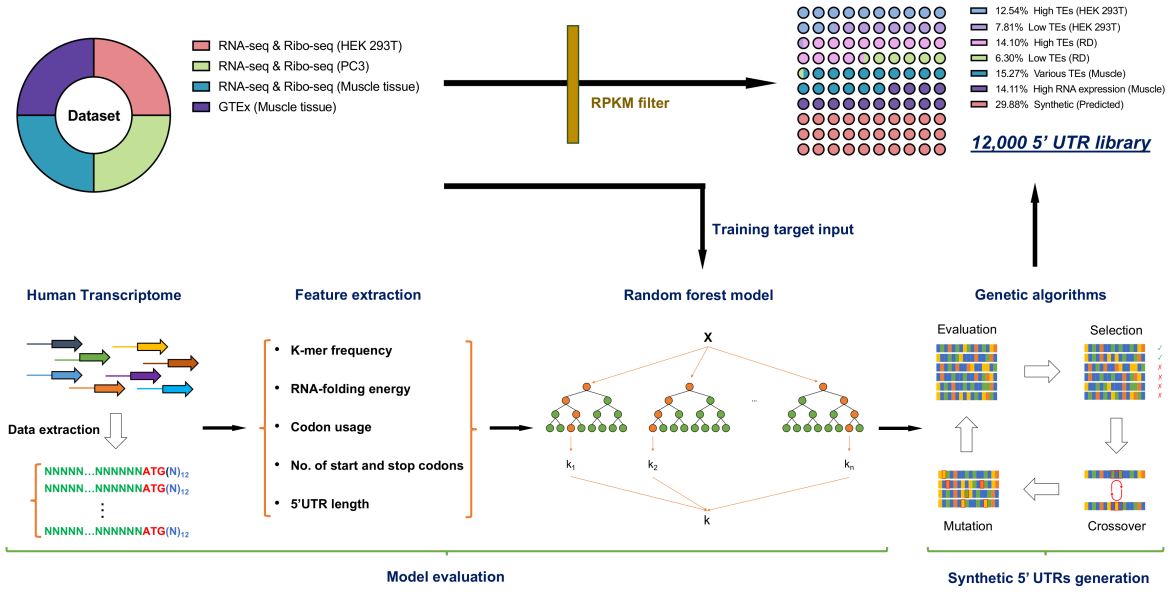


Figure 2. Design of the 5' UTR library of naturally occurring and synthetic 5' UTRs. RNA-seq and Ribo-seq datasets of HEK 293T, PC3, and human muscle cells, together with the GTEx database of human muscle tissue, were collected. Natural 5'UTRs with high TEs and low TEs in HEK 293T and RD cells, 5' UTRs with various TEs in human muscle cells, and the 5' UTRs with high mRNA counts in human muscle tissues were selected and added to the library. In addition, we designed synthetic 5' UTRs by: *i)* collecting endogenous 5' UTR sequences on the target cell type (HEK 293T, PC3 or human muscle cells) from public data; *ii)* extracting sequence features of the 5' UTRs, including those nucleotides surrounding the AUG region; *iii)* training a Random Forest machine learning method for each cell type/tissue (HEK 293T, PC3 or human muscle cells), to learn a function that maps sequence features to mRNA expression levels and TEs; and *iv)* designing a set of 100 bp synthetic sequences that are predicted to maximize TEs and protein expression levels using genetic algorithms.

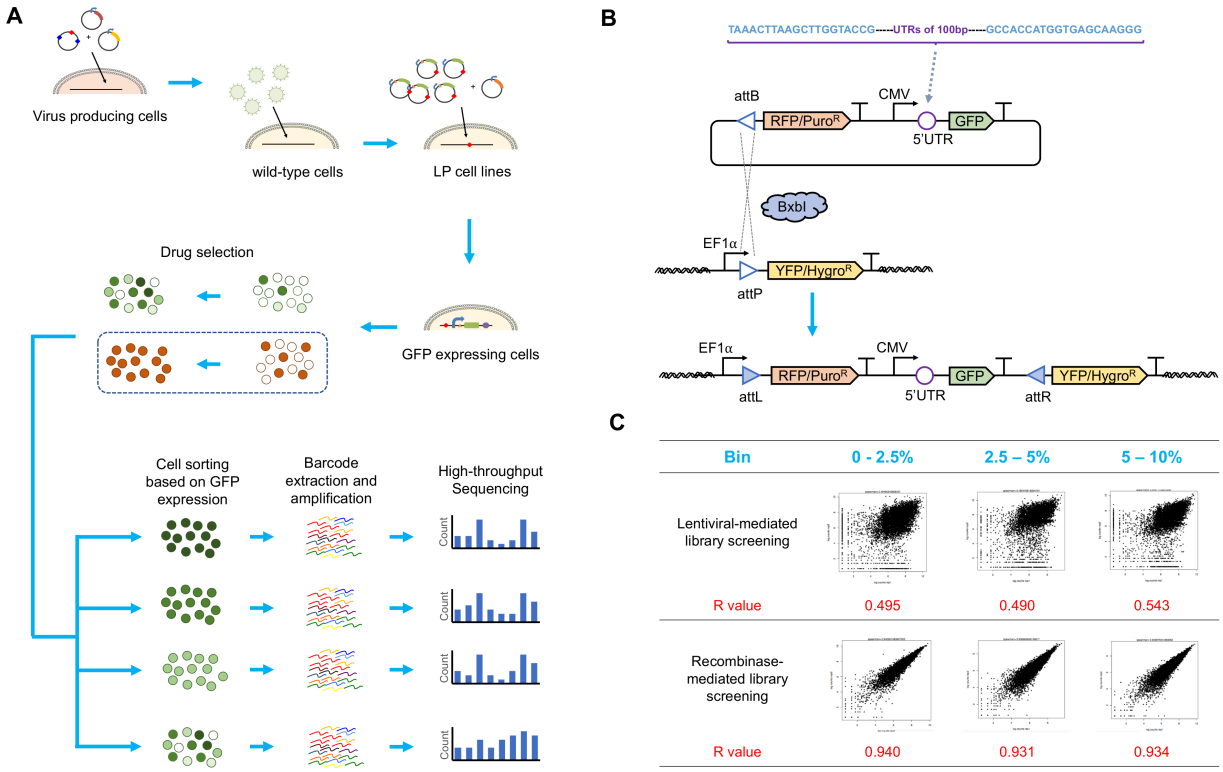


Figure 3. Strategy for constructing HEK 293T cell lines with a landing pad and screening the 5' UTR library using recombinase-based gene integration. A) Recombinase-based library screening workflow. B) Construction of the 5' UTR library and schematic illustration of recombinase-based gene integration. C) We observed high reproducibility for barcode representations between two HEK-LP cell lines independently transfected with the library and a recombinase-expression plasmid; cells were sorted into three bins based on GFP expression (top 0-2.5%, top 2.5-5%, and top 5-10%). log2 values of normalized barcode counts are shown. R is the Pearson correlation coefficient.

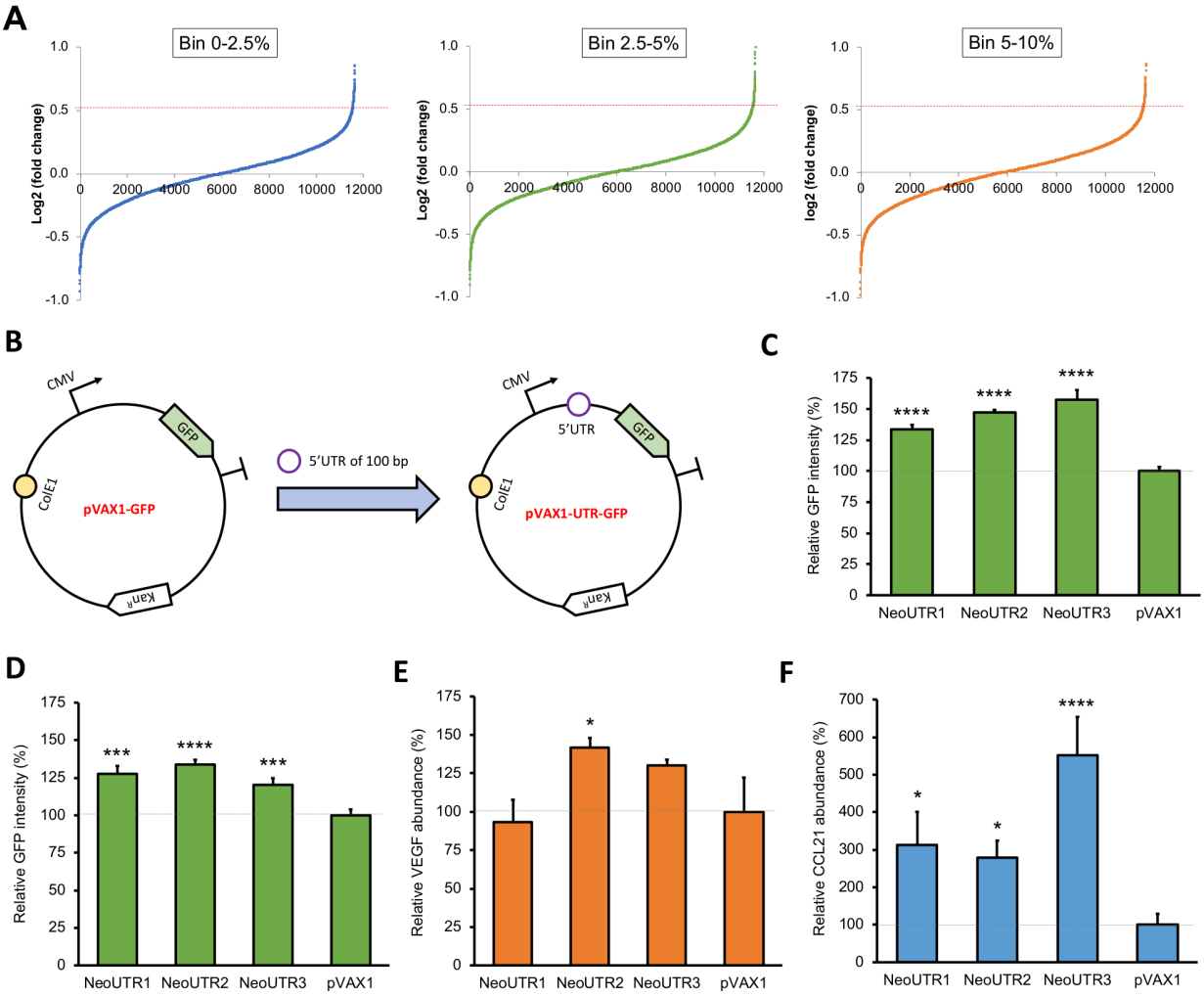


Figure 4. Selection and validation of 5' UTR candidates. A) 5' UTRs that modulate protein expression were ranked by their mean log2 ratios (compared with the control of unsorted cells) of the normalized barcode count in the three bins based on GFP expression. 5' UTRs with a log2 ratio greater than 0.52 (which is highlighted as a red dotted line) in all three bins were selected for further validation. B) The GFP gene was inserted into the pVAX1 plasmid to make the pVAX1-GFP plasmid, which was used as a control in the GFP expression study. 5' UTR candidates were inserted directly upstream of the Kozak sequence of the GFP coding sequence to make the pVAX1-UTR-GFP plasmids. C) Three 5' UTR candidates that significantly enhanced protein expression were chosen for further testing. D, E, F) The effects of the three 5' UTRs on GFP (D), VEGF (E), and CCL21 (F) expression in RD cells. Relative protein expression in each sample was normalized to that of the pVAX1 plasmid (relative expression = 1 is highlighted as a grey dotted line). Error bars indicate SD for three biological replicates. (*p<0.05; **p<0.01; ***p<0.001; ****p<0.0001 vs pVAX1)

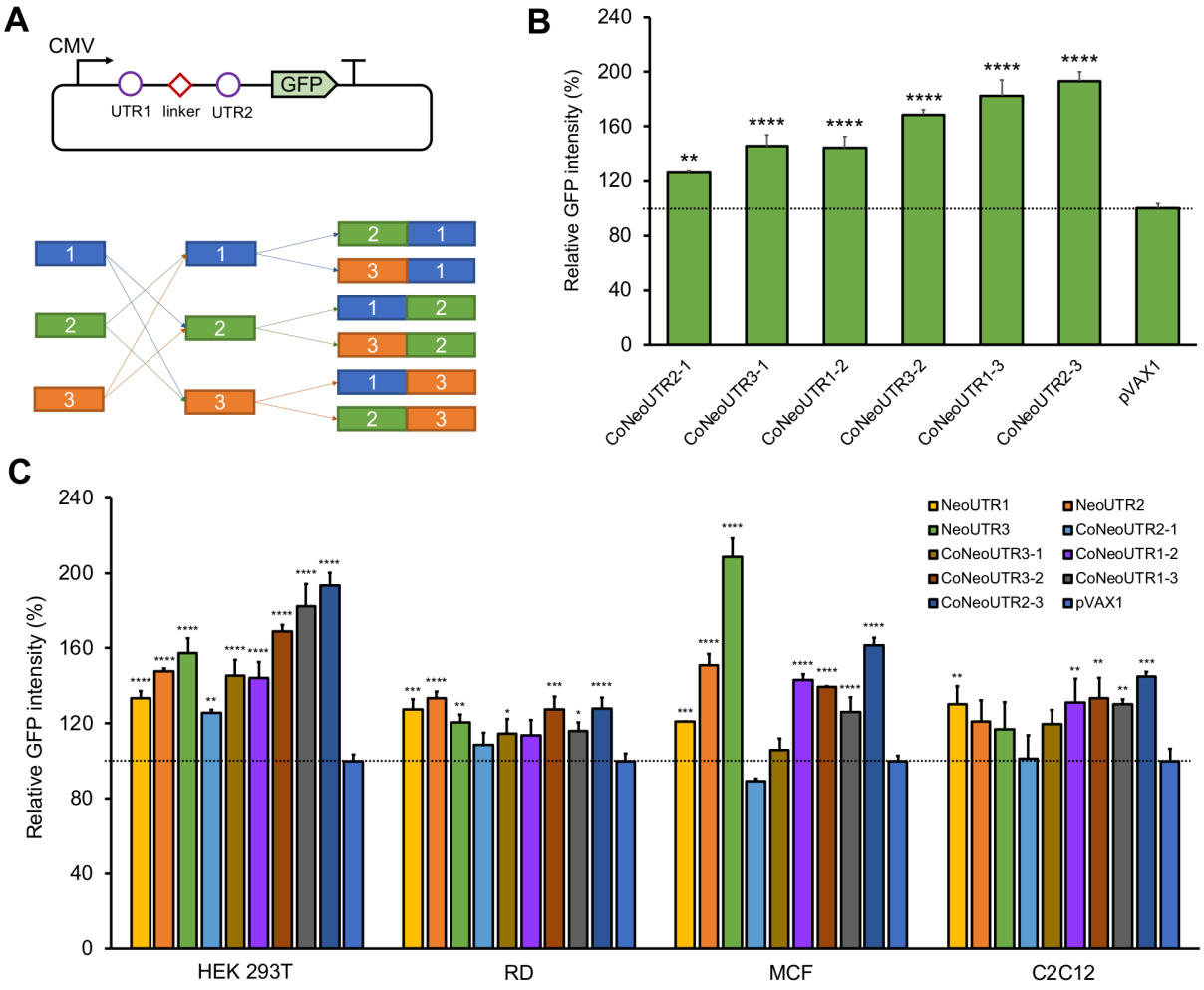


Figure 5. Effects of combinatorial 5' UTRs on GFP expression in various cell lines. A) We constructed six distinct 5' UTR combinations by combining different pairwise permutations of the three validated 5' UTR candidates with a CAACAA linker between them, and then inserted these combinations into the pVAX1-GFP plasmid directly upstream of the Kozak sequence. B) GFP expression from the 5' UTR combinations on GFP expression in HEK 293T cells. C) Test of the single and combinatorial 5' UTRs on GFP expression in various cell lines. The relative protein expression was normalized to that from the pVAX1-GFP plasmid, set as 1 and highlighted as a grey dotted line. Error bars indicate SD for three biological replicates. (*p<0.05; **p<0.01; ***p<0.001; ****p<0.0001 vs pVAX1)

SUPPORTING INFORMATION

High-Throughput 5' UTR Engineering for Enhanced Protein Production in Non-Viral Gene Therapies

Jicong Cao[1, 2, 3, 4, 7], Eva Maria Novoa[4, 5, 6, 7 #], Zhizhuo Zhang[4, 5, 6], William C.W. Chen[1, 2, 3], Dianbo Liu[1, 4, 5], Gigi C G Choi[1, 2, 3 *], Alan S L Wong[1, 2, 3 *], Claudia Wehrspaun[1, 2, 3], Manolis Kellis[4, 5, 6 †], Timothy K Lu[1, 2, 3 ,4, 6 †]

[1]*Synthetic Biology Group, Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*

[2]*Department of Biological Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*

[3]*Synthetic Biology Center, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*

[4]*Broad Institute of MIT and Harvard, Cambridge, MA 02142, USA.*

[5]*Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*

[6]*Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.*

[7]*The authors contributed equally to this work.*

[#]*Present address: Center for Genomic Regulation (CRG), 08003 Barcelona, Spain.*

[*]*Present address: School of Biomedical Sciences, University of Hong Kong, Hong Kong, China*

[†]*Corresponding authors: manoli@mit.edu, timlu@mit.edu*

CONTENTS

MATERIALS AND METHODS

1. 5' UTR library design and construction

To select the endogenous sequences, we used publicly available matched RNA-Seq and Ribo-Seq datasets, from three different human cell lines/tissues, which included: *i)* HEK 293T cells, *ii)* human prostate cancer (PC3) cells, and *iii)* human muscle tissue. The first two were chosen as these are commonly used cell lines, whereas the third human muscle tissue as it could be the target tissue of DNA vaccine therapy. We then analyzed the RNA-seq and Ribo-seq datasets, and determined their translation efficiency rates and mRNA levels. Per-transcript translation efficiency (TE) was defined as Ribo-seq RPKM/RNA-seq RPKM, where RPKM represents Reads Per Kilobase of transcript per Million mapped reads. Transcripts with insufficient RNA-Seq or Ribo-Seq coverage were discarded. Our final selection of nature-occurring 5' UTRs (**Supplementary Table 1)** consisted in: *i)* top 1505 sequences and bottom 937 sequences from transcripts with highest and lowest TEs from human embryotic kidney 293 (HEK293) cells, *ii)* top 1692 and bottom 756 sequences from transcripts with highest and lowest TEs from human prostate cancer (PC3) cells, *iii)* 1831 sequences from transcripts that displayed maximum TEs for muscle tissue, *iv)* 1693 5'UTR regions from transcripts with high mRNA expression levels in muscle tissue, which were extracted from publicly available data from the Genotype-Tissue Expression (GTEx) project.

Moreover, we trained and developed a computational model to predict the TE based on its 5'UTR characteristics. To establish this model, we first identified which sequence features of 5'UTR regions increased gene expression levels and TE. For this aim, we extracted several sequence features, including k-mer frequency, codon usage, RNA folding energy, 5'UTR length, and number of ORFs. We developed a computational model trained on sequence features to predict translation efficiency and mRNA expression on different cell conditions. The model was trained data on experimentally determined translation efficiency rates and mRNA levels, which were obtained from analyzing publicly available RNA-seq and Ribo-seq data of endogenous genes from three human cell types (HEK293 cells, PC3 cells, and human muscle tissue).

The workflow consisted on the following steps: *i)* Extract the sequences features of the 5'UTRs, including those nucleotides surrounding the AUG region, i.e. the whole 5'UTR plus 15 bp of the CDS sequences, for each of the expressed transcripts in each cell line or tissue. *ii)* Train a Random Forest machine learning method for each cell type/tissue, to learn a function that maps sequence features to mRNA expression and TE. *iii)* Design a set of 100 bp synthetic sequences that maximize TE and protein expression (where protein expression is computed as RNA levels * TE). Given that searching for all 4^{100} possible 100-bp sequences

would be too computationally demanding, we applied a genetic algorithm (GA), which simulates the evolution process, to search the optimal sequences by mutating and recombining the endogenous sequences. For each GA run, we randomly sampled 100 endogenous 5'UTR sequences as "initial population" prior to undergoing evolution and select them or their offspring based on their fitness, which is defined by their predicted TE or predicted Protein expression from the previous trained model. *iv)* From our GA results, we keep the top 5 sequences with at least 5 bp differences in each run. *v)* To validate the accuracy of our model, we also selected sequences with small number of mutations from its endogenous origin, but large increase/decrease of TE or Protein RNA expression comparing to its endogenous sequence. The first set was designed by evolving 1198 5' UTRs within 2 generations to test the algorithm (**Supplementary Table 2**). The second set of 2388 5' UTRs was designed as follows: from the output of each run, we took the best five sequences during the run, requiring them to have at least five mismatches of each other, and also requiring that their scores were at least 0.05 better than those of the initial naturally occurring sequences within a maximum of 50 generations (**Supplementary Table 3**).

A 12K oligonucleotide library of 140-mer was synthesized using CustomArray to contain 100 bp variable 5'end sequences flanked by PCR priming sites. The details of the library are in **Supplementary Table 7**. The library was cloned to the reporter plasmids using conventional restriction enzyme cloning and Gibson Assembly.

2. Plasmid construction

The plasmids used in this study were built using restriction enzyme cloning and Gibson assembly. All plasmid sequences used in this study are detailed in GenBank format in the single text file ''plasmid sequences.docx.''

3. The construction of the HEK 293T landing pad cell lines.

HEK 293T cells (ATCC, VA, USA) were grown in polystyrene flasks in Dulbecco's Modified Eagle's Medium (Life Technologies, CA, USA) supplemented with 10% fetal bovine serum (VWR, PA, USA) and 1% penicillin/streptomycin (Life Technologies, CA, USA) at 37 °C and 5% CO2. When the cells were 80-90% confluent, cells were harvested with 0.25% trypsin (Life Technologies, CA) for transfection. To make lentivirus containing the landing pad, HEK 293T cells were plated in 6-well plate format. In brief, 12 µL of FuGENE HD (Promega, WI, USA) was mixed with 100 µL of Opti-MEM medium Life Technologies, CA) and was added to a mixture of the three plasmids: 0.5 µg of lentiviral envelop vector pCMV-VSV-G

vector, 0.5 μg of lentiviral packaging vector psPAX2, and 1 μg of lentiviral expression vector for landing pad insertion pJC191 (**Supplementary Fig. 5**). After 20 minutes incubation of FuGENE HD/DNA complexes at room temperature, 1.8 million cells were added to each FuGENE HD/DNA complex tube, mixed well, and incubated for another 10 min at room temperature before being added to 6-well plates containing 1 mL cell culture medium, followed by incubation at 37°C and 5% CO2. The media was removed 24 hours after transfection and 2 mL fresh media was added. After another 24 hours transfection, supernatant containing newly produced viruses was collected, and filtered through a 0.45 mm syringe filter (Pall Corporation, MI, USA) and used for infection. The filtered supernatant was diluted by different titrations of viruses using fresh media, and mixed with 8 mg/mL polybrene before added into 6-well plates with 1 million cells seeded on each well 24 hours before infection. Cell culture medium was replaced the next day after infection and cells were cultured for at least 3 days prior to FACS analysis or sorting using BD FACSAria. Single YFP positive cells from the well with less than 10% YFP positive cells (roughly 0.1 MOI) were sorted into a 96-well plate and were culture in fresh medium for 2 weeks and expanded to 6 well plates with medium supplemented with 50 μg/mL hygromycin (Life Technologies, CA, USA). Two clones with single copy landing pad insertion, HEK-LP3 and HEK-LP9, were selected for as the parental landing pad cell lines for library screening.

4. Library transfection, recombinase-based library integration and next-generation sequencing.

The landing pad cells were seeded as 1 million per well on 6-well plated 24 hours before transfection. One μg library plasmid JC253L (**Supplementary Fig. 6**) carrying an attB site and the 5' UTR library and 1 μg BxbI recombinase expressing plasmid pCAG-BxbI were mixed with 6 uL FuGENE HD and added into each well. Eight wells of each landing pad cells were used for transfection. To ensure the reproducibility of our screening results, we maintained >25-fold coverage of each library member throughout the screening pipeline. 4 μg/mL puromycin was added three days post transfection and the cells were cultured for at least one more week. The cells were then analyzed using FACS and sorted into three bins based on distinct levels of GFP intensity while the unsorted cells were used as control.

For NGS library preparation, the genomic DNA was extracted from each bin and 800 ng were used as the template for PCR amplification with barcoded Pi7 primer. Sequencing was performed at the MIT BioMicro Center facilities on an Illumina NextSeq machine using 150 bp double-end reads.

5. Lentiviral-based screening of the 5' UTR library in HEK 293T cells.

When HEK 293T cells were 80-90% confluent, cells were harvested with 0.25% trypsin for transfection. For each well, 12 µL of FuGENE HD (Promega, WI, USA) was mixed with 100 µL of Opti-MEM medium Life Technologies, CA) and was added to a mixture of the three plasmids: 0.5 µg of lentiviral envelop vector pCMV-VSV-G vector, 0.5 µg of lentiviral packaging vector psPAX2, and 1 µg of lentiviral 5'UTR library plasmid pJC240L (**Supplementary Fig. 7**). After 20 minutes incubation of FuGENE HD/DNA complexes at room temperature, 1.8 million cells were added to each FuGENE HD/DNA complex tube, mixed well, and incubated for another 10 min at room temperature before being added to 6-well plates containing 1 mL cell culture medium, followed by incubation at 37°C and 5% CO2. The media was removed 24 hours after transfection and 2 mL fresh media was added. After another 24 hours transfection, supernatant containing newly produced viruses was collected, and filtered through a 0.45 mm syringe filter and used for infection.

The filtered supernatant was diluted by different titrations of viruses using fresh media, and mixed with 8 mg/mL polybrene before added into 6-well plates with 1 million HEK 293T cells seeded on each well 24 hours before infection. Cell culture medium was replaced the next day after infection and the infection efficiency were cultured for at least 3 days prior to FACS analysis using BD LSR II. The infected HEK 293T cells from the well with less than 10% GFP positive cells (roughly 0.1 MOI) were selected as the integration of a single copy of the 5' UTR was expected in most of the infected cells. To ensure the reproducibility of the screening results, we maintained >25-fold coverage of each library member throughout the screening pipeline. The infected cells were sorted and further expanded for at least one week. The cells were then analyzed using FACS and sorted into three bins based on distinct levels of GFP intensity while the unsorted cells were used as control.

For NGS library preparation, the genomic DNA was extracted from each bin and 800 ng were used as the template for PCR amplification with barcoded Pi7 primer. Sequencing was performed at the MIT BioMicro Center facilities on an Illumina NextSeq machine using 150 bp double-end reads.

6. NGS data pre-processing and analysis.

Fastq files were first inspected for quality control (QC) using FastQC. Fastq files were then filtered and trimmed using fastx_clipper of the FASTX-Toolkit. Trimmed fastq files were collapsed using fastx_collapser of the FASTX-Toolkit. The collapsed fasta file was used as an input for alignment in Bowtie2 with a very sensitive alignment mode and aligned against the library reference. The resulting SAM file was filtered for mapped reads using SAMtools, and the reads were then quantified by summing the

counts of each unique promoter using an in-house R script. The reads were normalized by dividing all reads in the sample by a size factor estimated by DESeq2. Replicability was assessed using Pearson correlation values using the *cor.test* function R. Differentially expressed 5'UTRs were identified using DESeq2. Top-ranked 5'UTRs, which were ranked based on their log2 fold change relative to the unsorted bin, were selected as *leads* for experimental validation.

7. Measurement of the GFP expression of the plasmids with 5' UTR candidates in mammalian cells.

HEK 293T cells, human rhabdomyosarcoma (RD) cells, human breast adenocarcinoma (MCF-7) Cells, and mouse C3H muscle myoblast (C2C12) cells were obtained from the American Type Culture Collection. HEK-293T, RD, MCF-7 and C2C12 cells were cultured in DMEM supplemented with 10% fetal bovine serum, and 1% Pen/Strep at 37ºC with 5% CO2. When the cells were 80-90% confluent, cells were harvested with 0.25% trypsin for transfection.

For HEK 293T cells, 50,000 cells per well on 96 well plates were plated 24 hours before the transfection and 50 ng of plasmid with different 5' UTRs or introns (**Supplementary Tables 8 and 9)** and 50 ng pEF1α-BFP was mixed with 0.3 uL FuGENE HD used in each well; For RD cells, 10,000 cells per well on 96 well plates were plated 24 hours before transfection and 50 ng of plasmid with pJC271 (**Supplementary Fig. 8**) or plasmids with different 5' UTRs and 50 ng pEF1α-BFP was mixed with 0.5 uL FuGENE HD used in each well; For MCF-7 cells, 20,000 cells per well on 96 well plates were plated 24 hours before transfection and 50 ng of plasmid with different 5' UTRs and 50 ng pEF1α-BFP was mixed with 0.5 uL FuGENE HD used in each well; For C2C12 cells, 10,000 cells per well on 96 well plates were plated 24 hours before transfection and 50 ng of plasmid with different 5' UTRs and 50 ng pEF1α-BFP was mixed with 0.3 uL Lipofectamine 2000 (Life Technologies, CA, USA) used in each well. After one or two days, the GFP and BFP intensity were measured using BD LSR II.

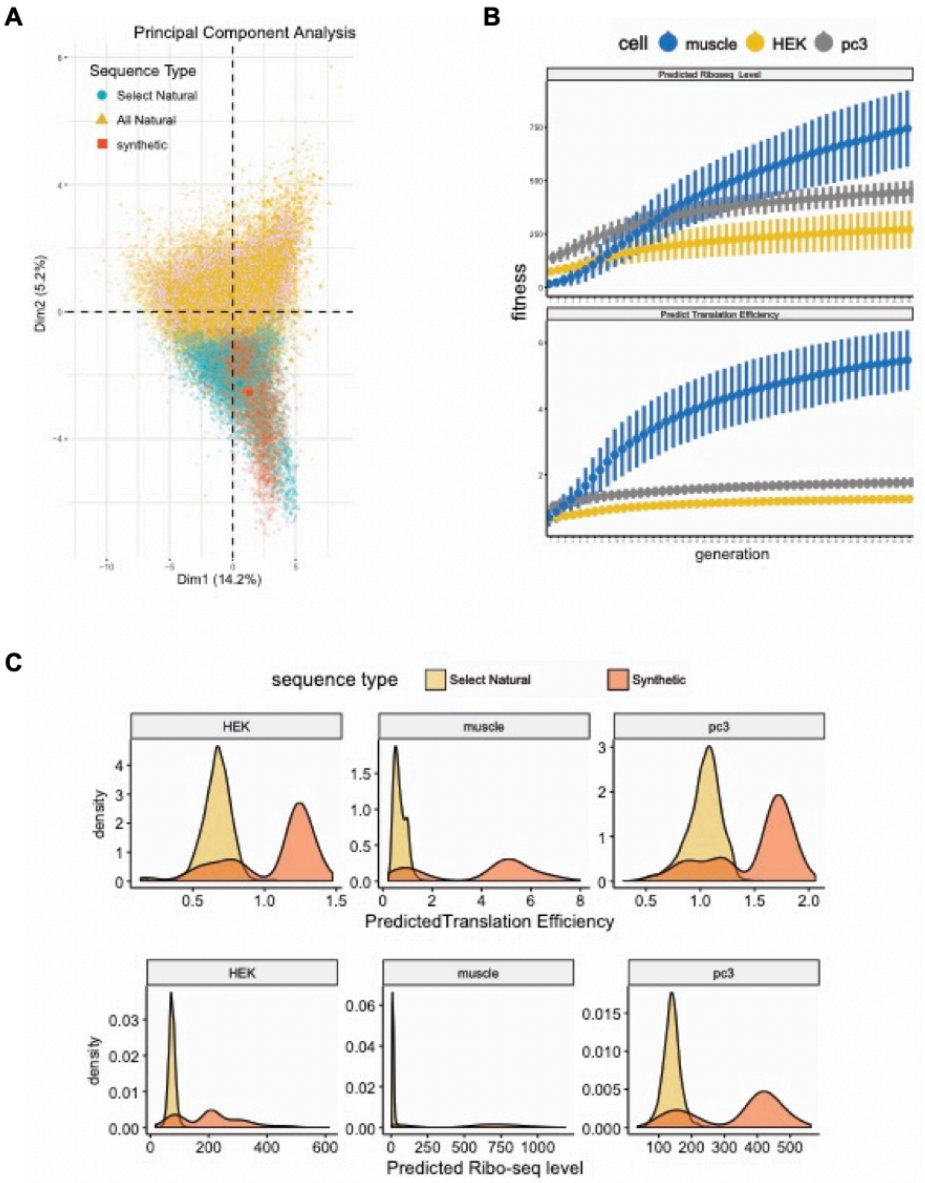
8. ELISA measurement of therapeutic protein production.

To determine productions of therapeutic proteins with 5' UTR candidates, we constructed plasmids encoding secretory human vascular endothelial growth factor (hVEGF) or C-C Motif Chemokine Ligand 21 (hCCL21), downstream of different 5' UTR candidates, respectively. HEK 293T cells were transfected with 100 ng of plasmid in 24-well plates at 100,000 cells per well and cultured for 24 hours with l mL complete culture medium. After washing cells once with PBS, complete culture medium was replaced with 0.5 mL plain DMEM supplemented with 1% Pen/Strep. Cells were incubated at 37ºC with 5% CO2 for

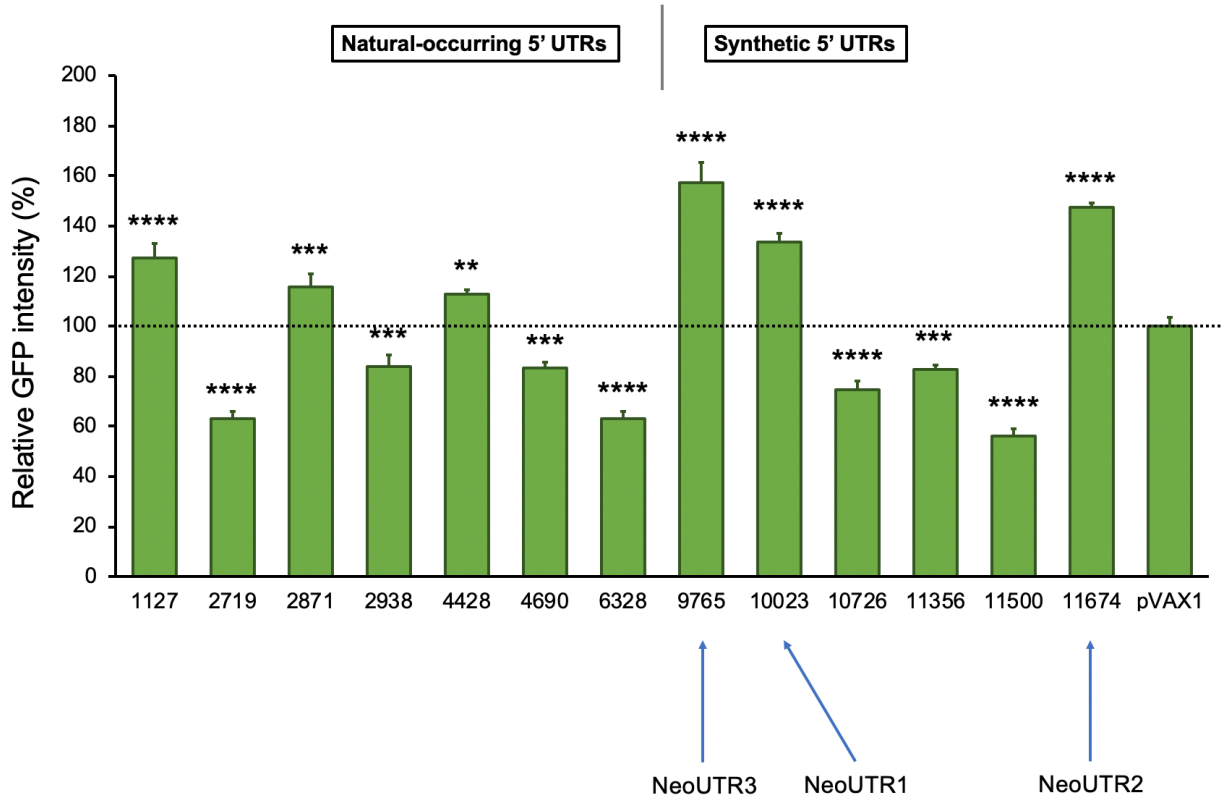
additional 24 hours. Supernatants were then collected, spun down at 350 g, and stored at -80ºC. The amount of each human protein in the supernatant was quantified by enzyme-linked immunosorbent assay (ELISA). Concisely, hVEGF concentration was determined by human VEGF ELISA Kit (KHG0111, Thermo Fisher Scientific), following the manufacturer's instructions; hCCL21 concentration was determined by human CCL21/6Ckine DuoSet ELISA (DY366, R&D systems), following the manufacturer's instructions. Data are presented as pg/ml per 100,000 cells per 24 hours.

9. Statistical Analysis

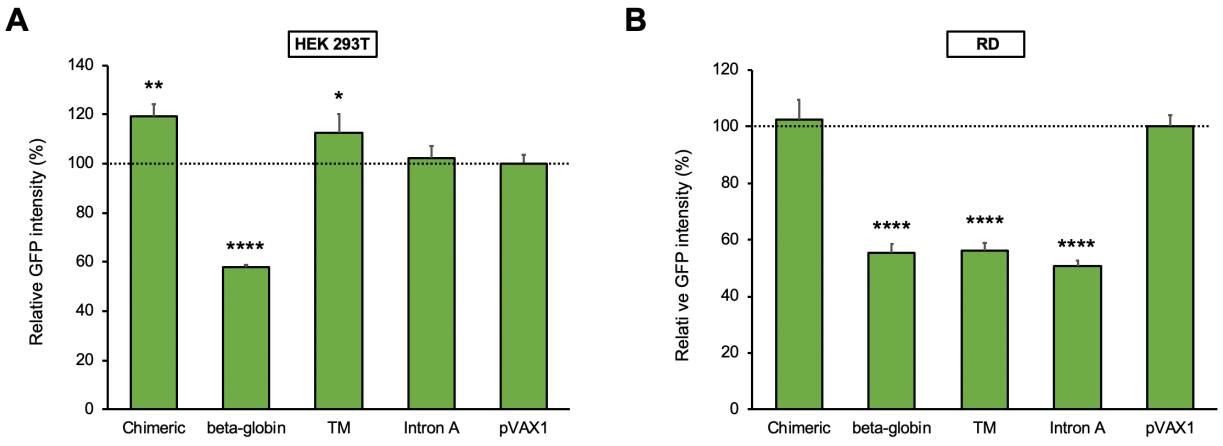
All quantitative data are presented as mean ± standard deviation (SD). Statistical differences between groups were analyzed by ordinary one-way ANOVA with 95% confidence interval. Statistical significance was set at $p \leqq 0.05$. Dunnett test was performed to correct for multiple comparisons in ANOVA post-hoc analysis. All statistical analyses were performed with GraphPad Prism 7.0 (GraphPad Software, La Jolla, CA, USA) statistics software.



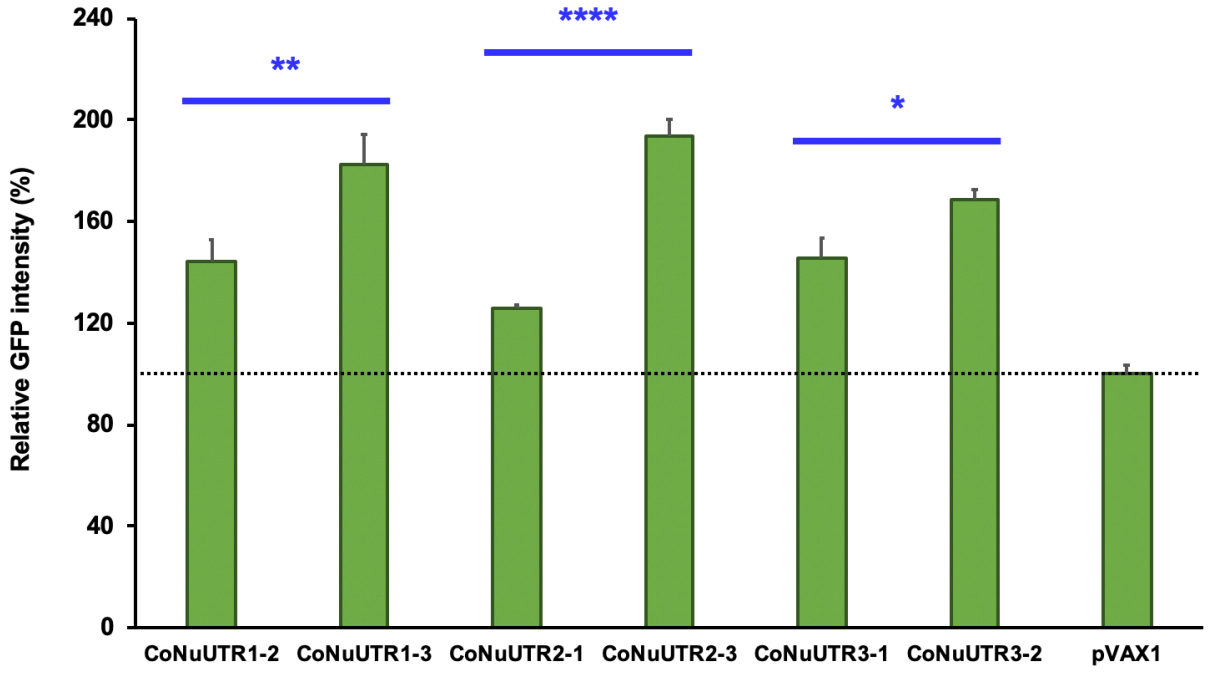
Supplementary Figure 1. The generation and characterization of the synthetic 5'UTRs. A) Principal component (PC) analysis of different type of sequence designs. The PCs were derived from all extracted features from the union set of UTR sequences. The scatter plot shows the first PC on the x-axis and the second PC on the y-axis. All naturally occurring human UTR sequences (orange), selected 8415 naturally occurring UTR sequences (green), 3586 synthetic UTR sequences. B) The fitness change of each generation during the GA. Upper panel is based on the Ribo-seq level prediction model, and the 2nd panel is based on the translation efficiency prediction model. C) The upper row shows the predicted translational efficiency (TE) distribution of selected natural UTRs (yellow) and synthetic UTRs (red) across three tested cell types (HEK, Muscle, PC3). The lower row shows Ribo-seq level distribution of selected natural UTRs (yellow) and synthetic UTRs (red) across three tested cell types (HEK, Muscle, PC3).



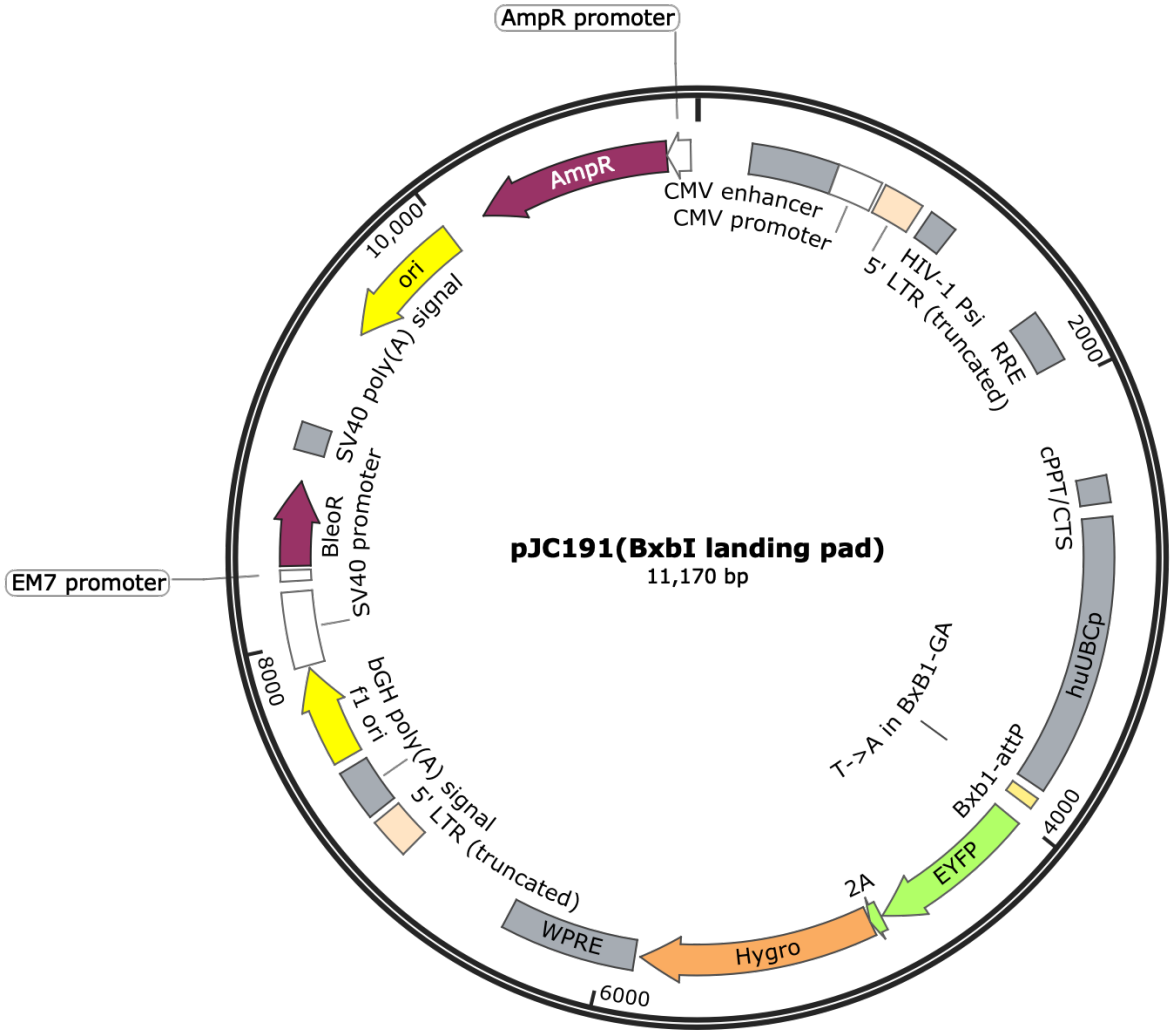
Supplementary Figure 2. The relative GFP intensities of the 13 candidate 5' UTRs in HEK 293T cells. Relative protein expression was normalized to that of the pVAX1-GFP plasmid, set as 1 and highlighted as a grey dotted line. Blue solid bars represent paired experimental groups. Error bars indicate SD for three biological replicates. (*p<0.05; **p<0.01; ***p<0.001; ****p<0.0001)



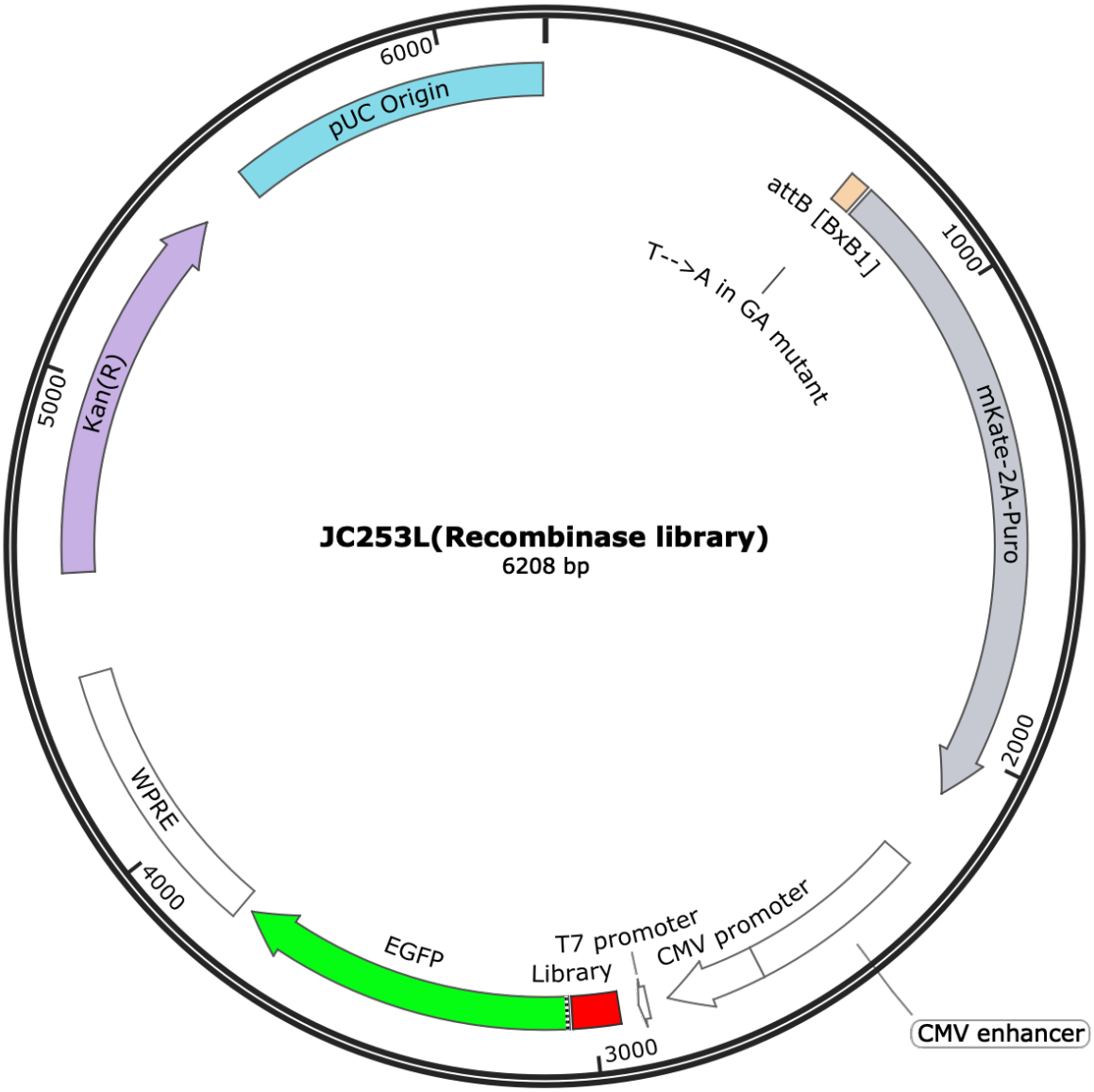
Supplementary Figure 3. The relative GFP intensities of the plasmids with different introns in HEK 293T (A) and RD (B) cells. Relative protein expression was normalized to that of the pVAX1-GFP plasmid, set as 1 and highlighted as a grey dotted line. Error bars indicate SD for three biological replicates. (*p<0.05; **p<0.01; ***p<0.001; ****p<0.0001 vs pVAX1)



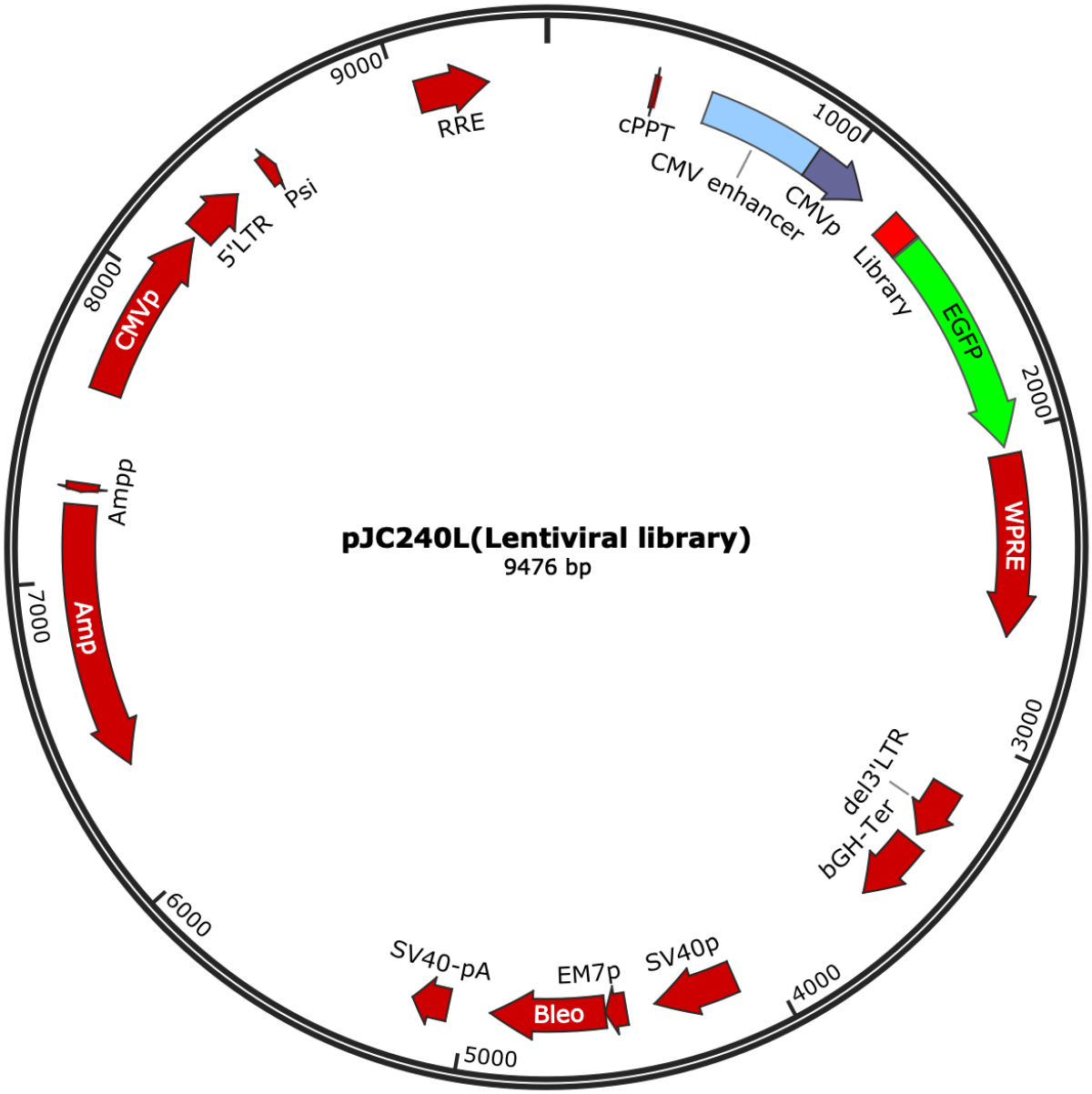
Supplementary Figure 4. The comparison of the fluorescence intensity between the combinatorial artificial UTRs. Relative protein expression was normalized to that of the pVAX1-GFP plasmid, set as 1 and highlighted as a grey dotted line. Error bars indicate SD for three biological replicates. (*p<0.05; **p<0.01; ***p<0.001; ****p<0.0001 vs pVAX1)



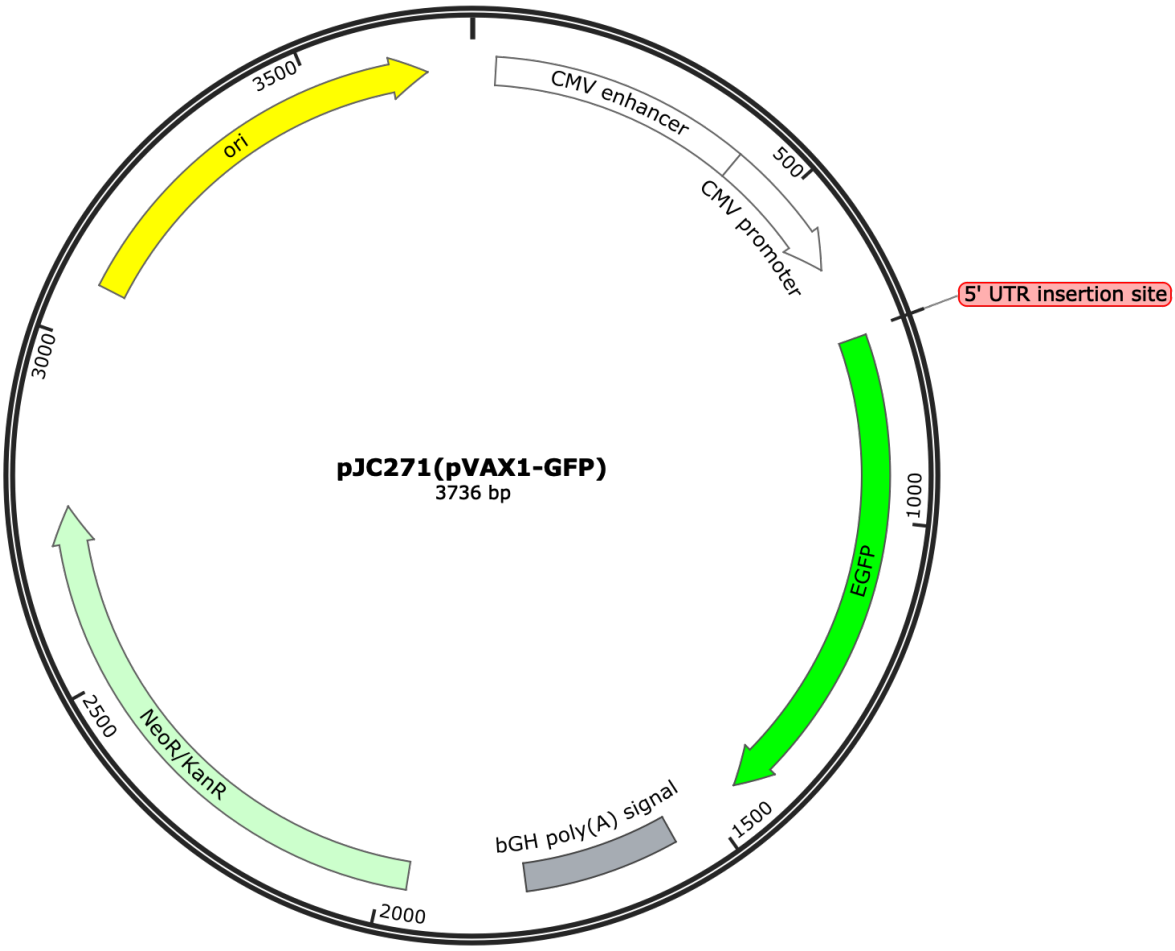
Supplementary Figure 5. Plasmid map of pJC191(BxbI landing pad)



Supplementary Figure 6. Plasmid map of pJC253L (Recombinase library)



Supplementary Figure 7. Plasmid map of pJC240L (Lentiviral library)



Supplementary Figure 8. Plasmid map of pJC271 (pVAX1-GFP)

Supplementary Tables

Supplementary Table 1. The sequences and characteristics of the selected naturally occurring 5' UTRs with flanking regions. (Attached as a separate excel file.)

Supplementary Table 2. The sequences and characteristics of the selected synthetic 5' UTRs with flanking regions. (Attached as a separate excel file.)

Supplementary Table 3. The sequences and characteristics of the selected synthetic 5' UTRs as a testing group with flanking regions. (Attached as a separate excel file.)

Supplementary Table 4. The NGS data analysis results of the 5' UTRs for the Bin 0-2.5%. (Attached as a separate excel file.)

Supplementary Table 5. The NGS data analysis results of the 5' UTRs for the Bin 2.5-5%. (Attached as a separate excel file.)

Supplementary Table 6. The NGS data analysis results of the 5' UTRs for the Bin 5-10%. (Attached as a separate excel file.)

Supplementary Table 7. The sequences of the ssDNA oligos used in this study. (Attached as a separate excel file.)

Supplementary Table 8. The sequences of the thirteen 5' UTR to validate.

Supplementary Table 9. The sequences of the four introns tested in this study.

Supplementary Table 8. The sequences of the thirteen 5' UTR to validate. UTR 10023 was named after as NeoUTR1, UTR 11674 was named after as NeoUTR2, UTR 9765 was named after as NeoUTR3.

UTR IDs	Sequences
1127	GCATTCCAACCTTCCAGCCTGCGACCTGCGGAGAAAAAAAATTACTTATTTTCTTGCC CCATACATACCTTGAGGCGAGCAAAAAAATTAAATTTTAACC
2719	CCACGGCTACTGCGTCCACGTGGCGGTGGCGTGGGGACTCCCTGAAAGCAGAGCGGC AGGGCGCCCGGAAGTCGTGAGTCGAGTCTTCCCGGGCTAATCC
2871	GCCGGTGGCGGCAGGATACAGCGGCTTCTGCGCGACTTATAAGAGCTCCTTGTGCGGC GCCATTTTAAGCCTCTCGGTCTGTGGCAGCAGCGTTGGCCCG
2938	CCTGCTGAAGGGGCCCGACTGGATCCTGGGCGAGATCAAGACATCGGGTTTGAGGGG CCGTGGAGGCGCTGGCTTCCCCACTGGCCTCAAGTGGAGCTTC
4428	CTACAGAAACGAAAGAAAAAGTCTGTATAAGCCAAAGGTGTTCGGGAAGAAAATAAC CCCATTGCCTTGAGTTTGTAGGTGCCACTACTACTCTGGAAAA
4690	ACCGGAAAGAGGGTGGCTGAGGTGGGGGAGGAGCCCAAAAGGCATTGTGGGAGTAC AGCTCTTTCCTTTCCGTCTGGCGGCAGCCATCAGGTAAGCCAAG
6328	CTGCCCGACAAAATACATCAGAATTTCTCTTTAAGAACAATATCGGATCGATTAAAAA ATATATATATCGGATCAAATTGGGGGTACTTCAATACCTTGC
9765	CTTGTCTCGCTCCGGGGAACGCTCGGAAACTCCCGGCCGCCGCCACCCGCGTCTGTTC TGTTACACAAGGGAAGAAAAGCCGCTGCCGCACTCCGAGTGT
10023	CATTCTGTGGTCTGATCATCCTGTGGTTTCGTCGCCGCCATCCTCGTCGCGACACGCTG TTTTCGGTTCTCGGCCCGACGAGCCATCGCCATCCTACAGC
10726	ACCAACAACCAACAACAACATCCACACCAACAACAACGCTGAAAGTGGTGTTTGCTTT CTCCACCAGAAGGGCACACTTTCATCTAATTTGGGGTATCGC
11356	CACCAGCTCCTCCACTCTCACACCCAGGATTCACAACCCAGGAGTCTAGACCCCCAGC CCCTCCACACTCCCACCCAGGAACAACCCGGATAGGTCGGAC
11500	GCACCACACCCGCTGCAACCAGCCCCTAGACCACTCACACACTGCACAGGGACCAGC AACAACAACAAGACTCTCACAGAGAGTCAGCCGGCCTTCATAG
11674	CACTCGCGCTGCCATCACTCTTCCGCCGTCTTCGCCGCCATCCTCGGCGCGACTCGCTT CTTTCGGTTCTACCAGGTAGAGTCCGCCGCCATCCTCCACC

Supplementary Table 9. The sequences of the four introns tested in this study.

Intron Names	Sequences
Chimeric	GTAAGTATCAAGGTTACAAGACAGGTTTAAGGAGGCCAATAGAAACTGGGCTTGTCGA GACAGAGAAGATTCTTGCGTTTCTGATAGGCACCTATTGGTCTTACTGACATCCACTTTG CCTTTCTCTCCACAG
Beta-globin	GTGAGTCTATGGGACCCTTGATGTTTTCTTTCCCCTTCTTTTCTATGGTTAAGTTCATGTC ATAGGAAGGGGAGAAGTAACAGGGTACACATATTGACCAAATCAGGGTAATTTTGCAT TTGTAATTTTAAAAAATGCTTTCTTCTTTTAATATACTTTTTTGTTTATCTTATTTCTAATA CTTTCCCTAATCTCTTTCTTTCAGGGCAATAATGATACAATGTATCATGCCTCTTTGCACC ATTCTAAAGAATAACAGTGATAATTTCTGGGTTAAGGCAATAGCAATATTTCTGCATAT AAATATTTCTGCATATAAATTGTAACTGATGTAAGAGGTTTCATATTGCTAATAGCAGCT ACAATCCAGCTACCATTCTGCTTTTATTTTATGGTTGGGATAAGGCTGGATTATTCTGAG TCCAAGCTAGGCCCTTTTGCTAATCATGTTCATACCTCTTATCTTCCTCCCACAG
TM	CGTTTAGTGAACCGTCAGATCCTCACTCTCTTCCGCATCGCTGTCTGCGAGGGCCAGCTG TTGGGCTCGCGGTTGAGGACAAACTCTTCGCGGTCTTTCCAGTACTCTTGGATCGGAAA CCCGTCGGCCTCCGAACGGTACTCCGCCACCGAGGGACCTGAGCGAGTCCGCATCGACC GGATCGGAAAACCTCTCGAGAAAGGCGTCTAACCAGTCACAGTCGCAAGGTAGGCTGA GCACCGTGGCGGGCGGCAGCGGGTGGCGGTCGGGGTTGTTTCTGGCGGAGGTGCTGCTG ATGATGTAATTAAAGTAGGCGGTCTTGAGACGGCGGATGGTCGAGGTGAGGTGTGGCA GGCTTGAGATCCAGCTGTTGGGGTGAGTACTCCCTCTCAAAAGCGGGCATTACTTCTGC GCTAAGATTGTCAGTTTCCAAAAACGAGGAGGATTTGATATTCACCTGGCCCGATCTGG CCATACACTTGAGTGACAATGACATCCACTTTGCCTTTCTCTCCACAGGTGTCCACTCCC AGGTCCAA
Intron A	GTAAGTACCGCCTATAGACTCTATAGGCACACCCCTTTGGCTCTTATGCATGCTATACTG TTTTTGGCTTGGGGCCTATACACCCCCGCTTCCTTATGCTATAGGTGATGGTATAGCTTA GCCTATAGGTGTGGGTTATTGACCATTATTGACCACTCCCCTATTGGTGACGATACTTTC CATTACTAATCCATAACATGGCTCTTTGCCACAACTATCTCTATTGGCTATATGCCAATA CTCTGTCCTTCAGAGACTGACACGGACTCTGTATTTTTACAGGATGGGGTCCCATTTATT ATTTACAAATTCACATATACAACAACGCCGTCCCCCGTGCCCGCAGTTTTTATTAAACAT AGCGTGGGATCTCCACGCGAATCTCGGGTACGTGTTCCGGACATGGGCTCTTCTCCGGT AGCGGCGGAGCTTCCACATCCGAGCCCTGGTCCCATGCCTCCAGCGGCTCATGGTCGCT CGGCAGCTCCTTGCTCCTAACAGTGGAGGCCAGACTTAGGCACAGCACAATGCCCACCA CCACCAGTGTGCCGCACAAGGCCGTGGCGGTAGGGTATGTGTCTGAAAATGAGCGTGG AGATTGGGCTCGCACGGCTGACGCAGATGGAAGACTTAAGGCAGCGGCAGAAGAAGAT GCAGGCAGCTGAGTTGTTGTATTCTGATAAGAGTCAGAGGTAACTCCCGTTGCGGTGCT GTTAACGGTGGAGGGCAGTGTAGTCTGAGCAGTACTCGTTGCTGCCGCGCGCGCCACCA GACATAATAGCTGACAGACTAACAGACTGTTCCTTTCCATGGGTCTTTTCTGCAG

SUPPLEMENTARY CODES

1. Extract sequences of 5'UTRs and CDSs (Save as "extractSequence_5utr_cds.R)

```
library(parallel)
library(doMC)
registerDoMC(cores=30)
library("BSgenome.Hsapiens.UCSC.hg19")
genome<-BSgenome.Hsapiens.UCSC.hg19
library(GenomicFeatures)
geneCacheFn="~/compbio/share_data/genomes/hg19db_GencodeV17.sqlite"

txdb <- loadDb(geneCacheFn)
cds<-unlist(cdsBy(txdb, by="tx", use.names=TRUE))
fiveUTRs<-unlist(fiveUTRsByTranscript(txdb,use.names=TRUE))

length(fiveUTRs)
###filter very short 5UTR ###
#filter=unlist(mclapply(fiveUTRs,function(x) width(ranges(x))>15))
#fiveUTRs=fiveUTRs[filter]
#length(fiveUTRs)

#get the first cds
match.cds=nearest(fiveUTRs,cds,select="all")
library(foreach)
topN=length(match.cds)
bed12format=foreach(k=1:topN,.combine=rbind)%dopar%{
i=match.cds[k]@queryHits
j=match.cds[k]@subjectHits
tx.id1=names(fiveUTRs[i])
tx.id2=names(cds[j])

#make sure they are the same transcript
if(tx.id1!=tx.id2){
return(NULL)
}

strand=as.character(strand(fiveUTRs[i]))

utr.r=ranges(fiveUTRs[i])
cds.r=ranges(cds[j])
##take first 15bp of CDS based on strand##
##order bed12 exon list based on strand ##
if(width(utr.r)+width(cds.r)<30){
        return(NULL)
}

chr=as.character(seqnames(fiveUTRs[i]))
```

```
start=min(start(utr.r), start(cds.r))
end=max(end(utr.r),end(cds.r))

firstNbase=15
if(strand=="+"){
        width(cds.r)=firstNbase
        blocksize.str=sprintf("%d,%d",width(utr.r) ,width(cds.r)  )
        blockstart.str=sprintf("%d,%d",utr.r@start-start, cds.r@start-start )
}else{
        start(cds.r)=start(cds.r)+width(cds.r)-firstNbase
        blocksize.str=sprintf("%d,%d",width(cds.r),width(utr.r))
        blockstart.str=sprintf("%d,%d",cds.r@start-start , utr.r@start-start)
}


#name=sprintf("%s__%d_%d",tx.id1,start,end)
c(chr,start,end,tx.id1,1,strand,start,end,"0,0,0",2,blocksize.str, blockstart.str  )

}


write.table(bed12format,"output/5utr_1stcds.bed12",sep="\t",col.names=F,quote=F,row.names=F)

cmd="fastaFromBed -fi ~/compbio/share_data/genomes/hg19.fa -bed output/5utr_1stcds.bed12  -name -s
-split -fo output/gencode_v17_5utr_15bpcds.fa"
system(cmd)
```

2. Preliminarily extract the features (Saved as "FeatureCommons.py")

```
####extract feature for latter prediction use
###RNA folding
###Condon
### k-mer
### motif
import sys,os
from Bio import SeqIO
import Bio.SeqUtils.CodonUsage
import subprocess
from multiprocessing import Pool
import gzip
from Bio.Seq import Seq

cds_length=15 ##assumption last portion of sequence is cds



def codonFreq(seq):
        codon_str=seq.translate().tostring()
        tot=len(codon_str)
        feature_map=dict()
        for a in codon_str:
                a="codon_"+a
                if a not in feature_map:
                        feature_map[a]=0
                feature_map[a]+=1.0/tot
        feature_map['uAUG']=codon_str.count("M") #number of start codon
        feature_map['uORF']=codon_str.count("*") #number of stop codon
        return feature_map

def singleNucleotide_composition(seq):
        dna_str=seq.tostring().upper()
        N_count=dict()  #add one pseudo count
        N_count['C']=1
        N_count['G']=1
        N_count['A']=1
        N_count['T']=1
        for a in dna_str:
                if a not in N_count:
                        N_count[a]=0
                N_count[a]+=1
        feature_map=dict()
        feature_map["CGperc"]=float(N_count['C']+N_count['G'])/len(dna_str)
        feature_map['CGratio']=abs(float(N_count['C'])/N_count['G']-1)
        feature_map['ATratio']=abs(float(N_count['A'])/N_count['T']-1)
        feature_map['utrlen_m80']=abs(len(dna_str)-80-cds_length)
        return feature_map
```

```
def RNAfold_energy(sequence, *args):
        rnaf = subprocess.Popen(["RNAfold","--noPS"] + list(args),
                        stdin=subprocess.PIPE,
                        stdout=subprocess.PIPE,
                        stderr=subprocess.PIPE,
                        # Universal Newlines effectively allows string IO.
                        universal_newlines=True)
        rnafold_output, folderr = rnaf.communicate(sequence)
    output_lines = rnafold_output.strip().splitlines()
    sequence = output_lines[0]
    structure = output_lines[1].split(None,1)[0].strip()
    energy = float(output_lines[1].rsplit("(",1)[1].strip("()").strip())
        return energy

def RNAfold_energy_Gquad(sequence, *args):
    rnaf = subprocess.Popen(["RNAfold","--noPS"] + list(args),
                    stdin=subprocess.PIPE,
                    stdout=subprocess.PIPE,
                    stderr=subprocess.PIPE,
                    # Universal Newlines effectively allows string IO.
                    universal_newlines=True)
    rnafold_output, folderr = rnaf.communicate(sequence)
    output_lines = rnafold_output.strip().splitlines()
    sequence = output_lines[0]
    structure = output_lines[1].split(None,1)[0].strip()
    energy = float(output_lines[1].rsplit("(",1)[1].strip("()").strip())
    return energy


def foldenergy_feature(seq):
        dna_str=seq.tostring()
        feature_map=dict()
        feature_map['energy_5cap']=RNAfold_energy(dna_str[:100])
        feature_map['energy_whole']=RNAfold_energy(dna_str)
        feature_map['energy_last30bp']=RNAfold_energy(dna_str[(len(dna_str)-30):len(dna_str)])
        feature_map['energy_Gquad_5utr']=RNAfold_energy_Gquad(dna_str[:(len(dna_str)-15)])
        feature_map['energy_Gquad_5cap']=RNAfold_energy_Gquad(dna_str[:50])
        feature_map['energy_Gquad_last50bp']=RNAfold_energy_Gquad(dna_str[(len(dna_str)-
50):len(dna_str)])
        return feature_map

def Kmer_feature(seq,klen=6):
        feature_map=dict()
        seq=seq.upper()
        for k in range(1,klen+1):
                for st in range(len(seq)-klen):
                        kmer=seq[st:(st+k)]
                        featname="kmer_"+kmer.tostring()
                        if featname not in feature_map:
                                feature_map[featname]=0
                        feature_map[featname]+=1.0/(len(seq)-k+1)
```

```
        return feature_map

def oss(cmd):
        print(cmd)
        os.system(cmd)


##seq is seq object from bio.python
def Seq2Feature(seq):
        ##codon
        ret=codonFreq(seq).items()
        ##DNA CG composition
        ret+=singleNucleotide_composition(seq).items()
        ##RNA folding
        ret+=foldenergy_feature(seq).items()
        ret+=Kmer_feature(seq).items()
        return ret
```

3. Extract the features (Saved as "FeatureExtraction_final.py")

```
####extract feature for latter prediction use
###RNA folding
###Condon
### k-mer
### motif
import sys,os
from Bio import SeqIO
import Bio.SeqUtils.CodonUsage
import subprocess
from multiprocessing import Pool
import gzip
from FeatureCommons import *

cds_length=15 ##assumption last portion of sequence is cds

#inputFasta="output/test.fa"
inputFasta=sys.argv[1]

###take the longest length for the same transcript id
tx_seq=dict()
for seq_record in SeqIO.parse(inputFasta, "fasta"):
        tx=seq_record.id
        seq=seq_record.seq
        if len(seq)<30: #skip when it too short
                continue
        if "ATG" not in seq:
                continue
        if tx not in tx_seq or len(seq)>len(tx_seq[tx]):
                tx_seq[tx]=seq


###output the non-redundancy fasta
outputFasta=inputFasta+".filter.fa"
outf=open(outputFasta,"w")
txIDlist=list()
seqList=list()
for tx in tx_seq:
        outf.write(">"+tx+"\n")
        outf.write(tx_seq[tx].tostring()+"\n")
        txIDlist.append(tx)
        seqList.append(tx_seq[tx])
outf.close()




pool = Pool(25)
featList=pool.map(Seq2Feature,seqList)
```

```
##output feature matrix, tx.id, feature.id, feature.value, id is 0-based
outf2=gzip.open(inputFasta+".sparseFeature.txt.gz",'wb')
feat2ID=dict()
featid=-1
for i in range(len(txIDlist)):
	txid=i
	for featItem in featList[i]:
		featname=featItem[0]
		featVal=featItem[1]
		if featname not in feat2ID:
			featid+=1
			feat2ID[featname]=featid
			fid=featid
		else:
			fid=feat2ID[featname]
		outstr=str(i)+"\t"+str(fid)+"\t"+str(featVal)
		outf2.write(outstr+"\n")

outf2.close()

##mapping id to human understandable name
outf3=open(inputFasta+".sparseFeature.rowname",'w')
for i in range(len(txIDlist)):
	outf3.write(str(i)+"\t"+txIDlist[i]+"\n")

outf3.close()

outf4=open(inputFasta+".sparseFeature.colname",'w')
sorted_items=sorted(feat2ID.items(), key=lambda x: x[1])
for a in sorted_items:
	print a
	featname=a[0]
	fid=a[1]
	outf4.write(str(fid)+"\t"+featname+"\n")
outf4.close()
```

5. Build the models using random forest (Save as "buildModel_final.R")

```
library(methods)
library(Matrix)
library(foreach)
library(doMC)
library(Metrics)
registerDoMC(cores=30)
prefix="output/gencode_v17_5utr_15bpcds.fa"
feat.df=read.table(sprintf("%s.sparseFeature.txt.gz",prefix))
feat.mat=sparseMatrix(i=feat.df[,1],j=feat.df[,2],x=feat.df[,3],index1=F)
rownames(feat.mat)=read.table(sprintf("%s.sparseFeature.rowname",prefix),row.names=1)[,1]
colnames(feat.mat)=read.table(sprintf("%s.sparseFeature.colname",prefix),row.names=1)[1:ncol(feat.mat)
,1] ##because last few Feature(kmer) may not occurs in the data, so need to trim

cell=".HEK_Andrev2015" ##HEK cell
#cell=".pc3"
#cell=""  ## muscle

#TE.df=read.table("data/df_counts_and_len.TE_sorted.with_annot.txt",row.names=1,header=T)

TE.df=read.table(sprintf("data/df_counts_and_len.TE_sorted%s.with_annot.txt",cell),row.names=1,heade
r=T)

if(cell==""){
mRNA.RPKM.filter=TE.df[,'rpkm_rnaseq']>5
ribo.RPKM.filter=TE.df[,'rpkm_riboseq']>0.1
}else{
mRNA.RPKM.filter=TE.df[,'rpkm_rnaseq']>50
ribo.RPKM.filter=TE.df[,'rpkm_riboseq']>5
}
nomiss=complete.cases(TE.df)
TE.df=TE.df[rownames(TE.df) %in% rownames(feat.mat) & mRNA.RPKM.filter & nomiss,]

sel.row=match(row.names(TE.df),rownames(feat.mat))

selfeat.mat=feat.mat[sel.row,]
te=TE.df[,'te']
ribo=TE.df[,'rpkm_riboseq']
rnaseq=TE.df[,'rpkm_rnaseq']

print(sprintf("Number of training samples: %d",length(te)))
library(e1071)
library(glmnet)
library(class)
library(randomForest)
library(rpart)
library(caret)


predictive_performance<-function(y){
```

```
model=2  ##random forest seems the best
#model=1 ## glmnet
test.id.list=createFolds(y,k=10)

featRange=1:ncol(selfeat.mat) #31

########Cross validation to evaluate different algorithm performance ##########
performances=foreach(i=1:length(test.id.list))%dopar%{
test.id=unlist(test.id.list[i])
train.x=selfeat.mat[-test.id,featRange]
train.y=y[-test.id]
test.x=selfeat.mat[test.id,featRange]
test.y=y[test.id]

nfolds=5
if(model==1){
#########glmnet###############
##train a model
fit=cv.glmnet(train.x,train.y,nfolds=nfolds,parallel=T,alpha=0)
pred.y=predict(fit$glmnet.fit,newx=test.x,s=fit$lambda.min)
}
if(model==2){
#########random forest#######
fit=randomForest(as.matrix(train.x), train.y)
pred.y=predict(fit,test.x)
}
if(model==3){
#######regression tree#######
df=data.frame(y=train.y,as.matrix(train.x))
fit=prune(rpart(y ~ .,df),cp=0.01)
pred.y=predict(fit,newdata=data.frame(y=NA,as.matrix(test.x)))
}
if(model==4){
fit=svm(train.x,train.y)
pred.y=predict(fit,test.x)
}

##evaluate
#cor(pred.y,test.y,method="spearman")
cor(pred.y,test.y)
}
print(mean(na.omit(unlist(performances))))
print(length(y))
}



### build full model and save ###
###model translation efficiency
print("building TE model.....")
y=log(te)
```

```
predictive_performance(y)

full.model.te=randomForest(as.matrix(selfeat.mat),log(te),importance=T)
print("Top features for predicting Translation Efficiency")
featScore=importance(full.model.te, type=1)
featScore[order(-featScore)[1:10],]
featScore=importance(full.model.te, type=2)
featScore[order(-featScore)[1:10],]

###model gene expression
print("building RNA expression model.....")

y=log(rnaseq)
predictive_performance(y)

full.model.rna=randomForest(as.matrix(selfeat.mat),log(rnaseq),importance=T)
print("Top features for predicting mRNA expression")
featScore=importance(full.model.rna, type=1)
featScore[order(-featScore)[1:10],]
featScore=importance(full.model.rna, type=2)
featScore[order(-featScore)[1:10],]

modelFn=sprintf("%s%s.big.model",prefix,cell)
save(full.model.te,full.model.rna,file=modelFn)
print(sprintf("save model to file : %s",modelFn))
```

5. Select the starting sequences for synthetic 5' UTRs generation (Save as "select_diverseDesginSequence.R")

```
similarThresh=160 #higher mean more similar
bestN=5
first3IterTop=1
first3IterBottom=1
library("seqinr")
library(Biostrings)
library(foreach)
library(doMC)
registerDoMC(cores=30)

Args<-commandArgs()[grep("^--",commandArgs(),invert=T)]
inputFn="output/final/raw/gencode_v17_5utr_15bpcds.fa.muscle.claudia_seq.galog.18870"
inputFn=Args[2]

options(stringsAsFactors = FALSE)

df=read.table(inputFn)
colnames(df)<-c("iter",'seq','score')

###prune the similar sequence
nearestSeq<-function(seq1,topSeqlist){
        if(is.null(topSeqlist)){
                return(0)
        }
        scores=foreach(ts=topSeqlist)%do%{
                pairwiseAlignment(seq1, ts,scoreOnly=T)
        }
        which.max(unlist(scores))
}



prune2<-function(seq1,topSeqlist){
    if(is.null(topSeqlist)){
        return(100)
    }
    scores=foreach(ts=topSeqlist)%do%{
        # pairwiseAlignment(seq1, ts,scoreOnly=T,type="overlap")
        unlist(adist(seq1,ts))
        }
    min(unlist(scores))
}



#####best N sequences must be better than the best endogenous sequence
selectBestNSequences<-function(){
topSeqlist=NULL
```

```
topSelID=NULL
bestInitScore=max(df[df[,'iter']==1,'score'])
df2=df[order(-df[,'score']),]
df2=df2[df2$score>bestInitScore,]
for(i in 1:nrow(df2)){
seq1=df2[i,'seq']

if(length(topSeqlist)>=bestN){
      break
}

if(prune2(seq1,topSeqlist)>5){
topSeqlist=c(topSeqlist,seq1)
topSelID=c(topSelID,i)
print(length(topSeqlist))
}

}

df2[unlist(topSelID),]
}

###bestN###
bestN.seqs=selectBestNSequences()

selectSimilarButBigChange<-function()
{
df1=df[df[,'iter']==1,] ##initial sequences
filter=df1[,'score']>-10
df1=df1[filter,]
df2=df[df[,'iter']==2|df[,'iter']==3,]
filter=df2[,'score']>-10
df2=df2[filter,]

scoreDlist=foreach(i=1:nrow(df2),.combine=rbind) %dopar%{
seq1=df2[i,'seq']
bestj=nearestSeq(seq1,df1[,'seq'])
scoreDiff=df2[i,'score']-df1[bestj,'score']
c(scoreDiff,bestj)
}
besti.top=which.max(scoreDlist[,1])
besti.bottom=which.min(scoreDlist[,1])
rbind(unlist(c(df2[besti.top,],sprintf("Up|%s|%f",df1[scoreDlist[besti.top,2],'seq'],df1[scoreDlist[besti.top,
2],'score'] ))),
unlist(c(df2[besti.bottom,],sprintf("Down|%s|%f",df1[scoreDlist[besti.bottom,2],'seq'],df1[scoreDlist[best
i.bottom,2],'score'] ))))
}

##similar but big change SBC
SBClist=selectSimilarButBigChange()
colnames(SBClist)[4]="description"
```

```
bestInitScore=max(df[df[,'iter']==1,'score'])

output.df=data.frame(bestN.seqs,description=rep(sprintf("bestTop|%f",bestInitScore),nrow(bestN.seqs)))
output.df=rbind(output.df,SBClist)

write.table(output.df,file=sprintf("output/final/sel/%s",basename(inputFn)),quote=F,col.names=F,sep="\t"
,row.names=F)
```

6. The design process of synthetic 5'UTRs based on TEs (Saved as "evolutionDesign_TE.R")

```
###use bigger training data
###smaller pop size, and fewer generations
###run more rounds


library(GA)
library(randomForest)
library(methods)
library(Matrix)
library(foreach)
library(doMC)
library(Metrics)
library(seqinr)
options(stringsAsFactors = FALSE)

registerDoMC(cores=30)

prefix="output/gencode_v17_5utr_15bpcds.fa"

cell=".HEK_Andrev2015" ##HEK cell
cell=".pc3"
#cell=""  ## muscle

Args<-commandArgs()[grep("^--",commandArgs(),invert=T)]
cell2=Args[2]
if(cell2=="muscle"){
cell=""
}else{
cell=paste(".",cell2,sep="")
}

TE.df=read.table(sprintf("data/df_counts_and_len.TE_sorted%s.with_annot.txt",cell),row.names=1,heade
r=T)
if(cell==""){
mRNA.RPKM.filter=TE.df[,'rpkm_rnaseq']>5
ribo.RPKM.filter=TE.df[,'rpkm_riboseq']>0.1
}else{
mRNA.RPKM.filter=TE.df[,'rpkm_rnaseq']>50
ribo.RPKM.filter=TE.df[,'rpkm_riboseq']>5
}


nomiss=complete.cases(TE.df)
TE.df=TE.df[ mRNA.RPKM.filter & nomiss,]



design_utr_len=100
```

```
if(cell!=""){
cell=paste(cell,".big",sep="")
}
print(cell)
print(load(sprintf("%s%s.model",prefix,cell)))

ACGT=1:4
names(ACGT)=c("A","C","G","T")

###Kozak-EGFP GCCACC + 15bp cds
GFPcds="GCCACCATGGTGAGCAAGGGC"
flank1='TAAACTTAAGCTTGGTACCG'
featureExtraction<-function(seq){
cmd=sprintf("python  -W ignore
FeatureExtraction_singleInput.py %s.sparseFeature.colname %s",prefix,paste(flank1,seq,GFPcds,sep="")
)
data <- (read.table(pipe(cmd),sep=" ",header=F,comment.char=""))
return(unlist(data[1,]))
}


predict_TE<-function(featVec){
predict(full.model.te,featVec)
}




##ACGT => 1234
DNA2intVec<-function(seq){
unlist(lapply(unlist(strsplit(toupper(seq),"")),function(x) ACGT[x]))
}

##1234 => ACGT
intVec2DNA<-function(vec){
paste(unlist(lapply(vec,function(x) names(ACGT)[x])),collapse="")
}

######generate initial by high TE 5utr#####
fiveUTR_Population<-function(object){
popSize=object@popSize
cdslen=20
##use Claudia generated sequence as initial pool
raw.seqlist=read.table(file ="output/final_endogenous.txt")[,1]

len.list=unlist(lapply(raw.seqlist,function(x) nchar(x)-cdslen))
prob.list=len.list/sum(len.list)  ##give high chance to the longer UTR sequence
population.seq=sample(raw.seqlist, size=popSize, replace = T, prob = prob.list)
population=foreach(seq = population.seq,.combine=rbind)%do%{
seq=substr(seq,1,nchar(seq)-cdslen)
```

```
seq=gsub("N","",seq,ignore.case=T)
seq_design=substr(seq,21,nchar(seq))
DNA2intVec(seq_design)
}
 return(as.matrix(population))
}


Endogenous_maxFitness<-function(){
        raw.seqlist=read.table(file ="output/final_endogenous.txt")[,1]
        TE.list=foreach(seq=raw.seqlist)%dopar%{
        cmd=sprintf("python  -W ignore
FeatureExtraction_singleInput.py %s.sparseFeature.colname %s",prefix,seq)
        data <- (read.table(pipe(cmd),sep=" ",header=F,comment.char=""))
        featVec=unlist(data[1,])
      predict_TE(featVec)
        }
        max(unlist(TE.list))
}

fitness<-function(intVec){
        utr.seq=intVec2DNA(intVec)
        featVec=featureExtraction(utr.seq)
        ###avoid ATG in the UTR sequences
        if(grepl("ATG",utr.seq)||grepl("AT$", utr.seq)){
                return(-10)
        }
        predict_TE(featVec)
}


gaMutation_ACGT<-function(object, parent ){
   mutate <- parent <- as.vector(object@population[parent, ])
   n <- length(parent)
   j <- sample(1:n, size = 1)
   alphabet=1:4
   mutate[j] <- sample(alphabet[-mutate[j]],1)
   return(mutate)
}

randkey=sample(1:100000, 1)

gaMonitor_writelog<-function(object, digits = getOption("digits")){
logFile=sprintf("output/final/raw/%s.%s.final.galog.%d",basename(prefix),cell2,randkey)
  fitness <- na.exclude(object@fitness)
   cat(paste("Iter =", object@iter, " | Mean =", format(mean(fitness),
      digits = digits), " | Best =", format(max(fitness), digits = digits),
      "\n"))
###write log file ###
##iter, seq, fitness
sink(logFile,append=T)
```

```
for(i in 1:nrow(object@population)){
        seq=intVec2DNA(object@population[i,])
        TE=object@fitness[i]
        cat(paste(object@iter,seq,TE,"\n"))
}
sink()
}

GA<-ga(type = "binary", fitness = fitness,nBits=design_utr_len ,min = 1, max = 4,maxiter=50,
popSize=100,population=fiveUTR_Population,parallel=T,monitor=gaMonitor_writelog,mutation=gaMut
ation_ACGT,seed=randkey )
```

7. The design process of synthetic 5'UTRs based on ribosome abundances (Saved as "evolutionDesign_Ribo.R")

```
###use bigger training data
###smaller pop size, and fewer generations
###run more rounds


library(GA)
library(randomForest)
library(methods)
library(Matrix)
library(foreach)
library(doMC)
library(Metrics)
library(seqinr)
options(stringsAsFactors = FALSE)

registerDoMC(cores=30)

prefix="output/gencode_v17_5utr_15bpcds.fa"

cell=".HEK_Andrev2015" ##HEK cell
cell=".pc3"
#cell=""  ## muscle

Args<-commandArgs()[grep("^--",commandArgs(),invert=T)]
cell2=Args[2]
if(cell2=="muscle"){
cell=""
}else{
cell=paste(".",cell2,sep="")
}

TE.df=read.table(sprintf("data/df_counts_and_len.TE_sorted%s.with_annot.txt",cell),row.names=1,heade
r=T)
if(cell==""){
mRNA.RPKM.filter=TE.df[,'rpkm_rnaseq']>5
ribo.RPKM.filter=TE.df[,'rpkm_riboseq']>0.1
}else{
mRNA.RPKM.filter=TE.df[,'rpkm_rnaseq']>50
ribo.RPKM.filter=TE.df[,'rpkm_riboseq']>5
}


nomiss=complete.cases(TE.df)
TE.df=TE.df[ mRNA.RPKM.filter & nomiss,]



design_utr_len=100
```

```
if(cell!=""){
cell=paste(cell,".big",sep="")
}
print(cell)
print(load(sprintf("%s%s.model",prefix,cell)))

ACGT=1:4
names(ACGT)=c("A","C","G","T")

###Kozak-EGFP GCCACC + 15bp cds
GFPcds="GCCACCATGGTGAGCAAGGGC"
flank1='TAAACTTAAGCTTGGTACCG'

featureExtraction<-function(seq){
cmd=sprintf("python  -W ignore
FeatureExtraction_singleInput.py %s.sparseFeature.colname %s",prefix,paste(flank1,seq,GFPcds,sep="")
)
data <- (read.table(pipe(cmd),sep=" ",header=F,comment.char=""))
return(unlist(data[1,]))
}

##should be produce logRibo level
predict_TE<-function(featVec){
predict(full.model.te,featVec)+predict(full.model.rna,featVec)
}



##ACGT => 1234
DNA2intVec<-function(seq){
unlist(lapply(unlist(strsplit(toupper(seq),"")),function(x) ACGT[x]))
}

##1234 => ACGT
intVec2DNA<-function(vec){
paste(unlist(lapply(vec,function(x) names(ACGT)[x])),collapse="")
}


######generate initial by high TE 5utr#####
fiveUTR_Population<-function(object){
popSize=object@popSize
cdslen=20
##use Claudia generated sequence as initial pool
raw.seqlist=read.table(file ="output/final_endogenous.txt")[,1]

len.list=unlist(lapply(raw.seqlist,function(x) nchar(x)-cdslen))
prob.list=len.list/sum(len.list)  ##give high chance to the longer UTR sequence
population.seq=sample(raw.seqlist, size=popSize, replace = T, prob = prob.list)
```

```
population=foreach(seq = population.seq,.combine=rbind)%do%{
seq=substr(seq,1,nchar(seq)-cdslen)
seq=gsub("N","",seq,ignore.case=T)
seq_design=substr(seq,21,nchar(seq))
DNA2intVec(seq_design)
}
 return(as.matrix(population))
}



Endogenous_maxFitness<-function(){
        raw.seqlist=read.table(file ="output/final_endogenous.txt")[,1]
        TE.list=foreach(seq=raw.seqlist)%dopar%{
        cmd=sprintf("python  -W ignore
FeatureExtraction_singleInput.py %s.sparseFeature.colname %s",prefix,seq)
        data <- (read.table(pipe(cmd),sep=" ",header=F,comment.char=""))
        featVec=unlist(data[1,])
      predict_TE(featVec)
        }
        max(unlist(TE.list))
}

fitness<-function(intVec){
        utr.seq=intVec2DNA(intVec)
        featVec=featureExtraction(utr.seq)
        ###avoid ATG in the UTR sequences
        if(grepl("ATG",utr.seq)||grepl("AT$", utr.seq)){
                return(-10)
        }
        predict_TE(featVec)
}


gaMutation_ACGT<-function(object, parent ){
   mutate <- parent <- as.vector(object@population[parent, ])
   n <- length(parent)
   j <- sample(1:n, size = 1)
   alphabet=1:4
   mutate[j] <- sample(alphabet[-mutate[j]],1)
   return(mutate)
}

randkey=sample(1:100000, 1)

gaMonitor_writelog<-function(object, digits = getOption("digits")){
logFile=sprintf("output/final/raw/%s.%s.final.gaRibo.%d",basename(prefix),cell2,randkey)
  fitness <- na.exclude(object@fitness)
   cat(paste("Iter =", object@iter, " | Mean =", format(mean(fitness),
      digits = digits), " | Best =", format(max(fitness), digits = digits),
```

```
        "\n"))
###write log file ###
##iter, seq, fitness
sink(logFile,append=T)
for(i in 1:nrow(object@population)){
        seq=intVec2DNA(object@population[i,])
        TE=object@fitness[i]
        cat(paste(object@iter,seq,TE,"\n"))
}
sink()
}

GA<-ga(type = "binary", fitness = fitness,nBits=design_utr_len ,min = 1, max = 4,maxiter=50,
popSize=100,population=fiveUTR_Population,parallel=T,monitor=gaMonitor_writelog,mutation=gaMut
ation_ACGT,seed=randkey )
```

8. Compile and format the synthetic 5' UTR library (Saved as "finalFormat_3k_synthetic_seqs.py")

```
import os,sys
import glob
import math


allFiles=glob.glob("output/final/sel/*")


model_ostr_score=dict()
visited=set()
#gencode_v17_5utr_15bpcds.fa.pc3.galog.62355
#gencode_v17_5utr_15bpcds.fa.pc3.claudia_seq.gaRibo.13167#
Nprinted=0
print "seq	score	model	generation	info"
for fn in allFiles:
	label=os.path.basename(fn).replace("gencode_v17_5utr_15bpcds.fa.","").replace(".claudia_seq","
").replace(".gaRibo","_Ribo").replace(".galog","_TE").replace(".final","").split(".")[0]
	for line in open(fn):
		comps=line.strip().split()
		seq="TAAACTTAAGCTTGGTACCG"+comps[1]+"GCCACCATGGTGAGCAAGGG"
		if seq in visited:
			continue
		visited.add(seq)
		score=comps[2]
		if score=="NA":
			continue
		itera=comps[0]
		info=comps[3]
		outstr=seq+"\t"+score+"\t"+label+"\t"+itera+"\t"+info
		if info.startswith("best"):
			initBestScore=float(info.split("|")[1])
			if float(score)<initBestScore+0.05:
				continue
			if label not in model_ostr_score:
				model_ostr_score[label]=dict()
			model_ostr_score[label][outstr]=float(score)
		else:
			print outstr
			Nprinted+=1
Ntotal=3585
import operator
perModelNum=int(math.ceil(float(Ntotal-Nprinted)/len(model_ostr_score)))

for model in model_ostr_score:
	x=model_ostr_score[model]
	sorted_x = sorted(x.items(), key=operator.itemgetter(1),reverse=True)
	for i in range(perModelNum):
		print sorted_x[i][0]
```