Genomorientierte Bioinformatik

-

Differential Analysis

Malte Weyrich

11. Januar 2025

Alternatives Spleißing ist ein fundamentaler zellulärer Mechanismus mit großen Einfluss auf regulatorische Prozesse und Genprodukte. Dabei werden die *Exons* eines Transkripts teilweise übersprungen oder neu angeordnet. Verantwortlich für diesen Prozess ist das *Spleißosom*. Im Folgenden wird ein Programm zur Quantifizierung von *Skipped Exon Events* vorgestellt und dessen Ergebnisse mit einem Hypothesen Test evaluiert. Es wurde auf zehn *bam*-Dateien ausgeführt mit *annotation_b37.gtf* als Referenzgenom. Zusätzlich werden die Resultate mit einem bereist publizierten Tool (***DEXSeq*** aus Anders, Reyes, and Huber 2012) verglichen.

Inhalt

1 — Berechnung der Percent Spliced-In Werte

1.1. Definition

Percent Spliced-In (PSI) Werte werden in der Bioinformatik genutzt, um die Evidenz von *Skipped Exon Events* anhand von beobachteten Daten darzustellen. Die *Skipped Exon Events* entstehen durch *Alternatives Spleißen* und beeinflussen maßgeblich das Proteinendprodukt eines *Gens*. Die Expression von Transkripten kann mittels *RNAseq* untersucht werden, bei dem die Sequenzen der vorliegenden Transkripte sequenziert werden. Somit entstehen Milliarden von *Reads*, welche mit der Hilfe von *Software* zurück auf das *Referenz Genom* projiziert werden können. Anschließend werden die *Reads* annotiert, das heißt es wird geschaut, von welchen Transkripten die *Reads* stammen. Der *PSI* Wert wird dann mittels *Inclusion Read Counts (**IRC**)* und *Exclusion Read Counts (**ERC**)* berechnet werden:

$$PSI := \frac{IRC}{IRC + ERC}.$$

Sei G ein *Gen* mit Transkripten WT und SV. Zudem sei se eine, durch die Annotation implizierte, übersprungene Region in WT, für die der *PSI* Wert berechnet werden soll. Wenn R die Menge an aligniernten *Read Pairs* auf G ist, dann ist $I \subseteq R$ die Menge an alignierten *Read Pairs* auf se und somit

$$IRC_{se} := |I| .$$

Die *Exclusion Reads* $E \subseteq R$ sind genau die *Read Pairs*, welche nicht auf se *gemapped* werden können.

$$ERC_{se} := |E| .$$

Ein *PSI* Wert von 1 würde beispielsweise bedeuten, dass die Region se in den beobachteten Transkripten eines *Gens* überwiegend inkludiert wurde (also nicht durch das *Spleißosom* heraus gespließt wurde), während ein Wert von 0 auf ein vermehrtes Ausschließen von se hindeuten würde.

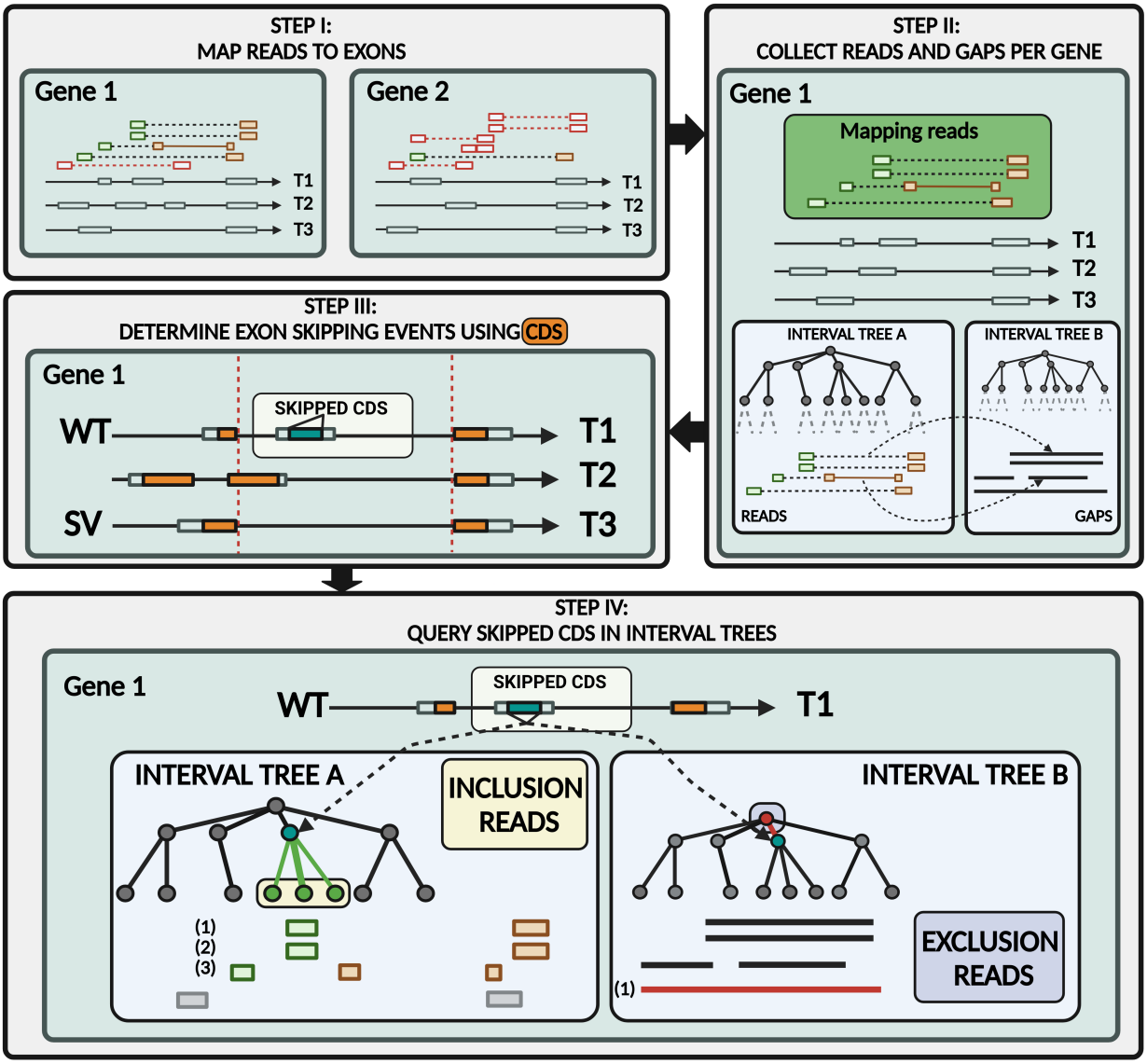

Abbildung 1 — Programmlogik zur Berechnung der *IRC* und *ERC counts* pro *Skipped Exon Event*. In dem Beispiel hätte das übersprungene Exon von G1 einen *PSI* Wert von 0.75. Die Abbildung wurde mit BioRender 2024 erstellt.

Der *JAR* werden eine *bam*- und ein *GTF*-Datei übergeben. Zudem muss eine Ausgabedatei Spezifiziert werden:

```
java -jar psi.jar -bam <bam>
                  -gtf <gtf>
                  -o <ausgabe.psi>
```

Die *JAR* liest zunächst die *GTF*-Datei ein und speichert alle *Gene* und deren annotierte *Transkripte* ab. Dabei enthält jedes *Transkript* jeweils dessen *Exons* **und** *Coding DNA Sequences (CDS)*. Gleichzeitig wird die *bam*-Datei mittels der *samtools library* decodiert und abgespeichert. Nach der Einleseroutine kann das Programm in vier Schritte eingeteilt werden (siehe Abbildung 1):

I. MAP READS TO EXONS

Die *Read Pair* Koordinaten werden mit den ***Exon*** Koordinaten der Transkripte abgeglichen. Sobald mindestens ein *Read Pair* auf ein *Gen mapped*, wird das *Gen* in einer *ArrayList<Gene>* ***mappedGenes*** abgelegt. Die Kriterien für einen validen *Read* wurden bereits im letzten Report geklärt. Es werden nur *Reads* in *mappedGenes* hinzugefügt, wenn sie auf genau ein einziges *Transkript* passen.

II. COLLECT READS AND GAPS PER GENE

Für jedes der Gene in ***mappedGenes*** werden nun zwei Intervallbäume A, B erstellt. Baum A beinhaltet alle *AlignmentBlocks* der zum Gen zugeteilten *Read Pairs*, während Baum B die Lücken zwischen einzelnen *AlignmentBlocks* und zwischen den *Forward* und *Reverse Reads* abspeichert.

III. DETERMINE EXON SKIPPING EVENTS USING CDS

Im nächsten Schritt wird über alle Gene aus ***mappedGenes*** iteriert. Für jedes einzelne Gen werden alle *Skipped Exon Events* mittels der *CDS* des Gens berechnet. Dabei wird dieselbe Logik wie in Report 1 verwendet.

IV. QUERY SKIPPED CDS IN INTERVALTREE

Für jede *CDS* werden nun dessen Intervalle in den Bäumen A, B abgefragt und die *Read IDs* gezählt.

$$IRC := |\{A.\textbf{\textit{getIntervalsSpannedBy}}(CDS.getStart(), CDS.getStop())\}|$$

$$ERC := |\{B.\textbf{\textit{getIntervalsSpanning}}(CDS.getStart(), CDS.getStop())\}|$$

Dabei werden für IRC **Intervalle** in A betrachtet, **die von der *CDS* überspannt werden** ($\equiv$ "*Reads* die innerhalb der *CDS* liegen") und für ERC werden **Intervalle** in B gesucht, **die die *CDS* überspannen** ($\equiv$ "Intervalle, die die *CDS* beinhalten").

Die *PSI* Werte werden in die Ausgabedatei mit folgendem Format geschrieben:

```
Gene ID            cdsStart-cdsStop    IRC   ERC  Total  PSI
ENSG00000165623.5  13275735-13275801   25    13   38     0.65
...                ...                 ...   ...  ...    ...
```

2 — Hypothesen Test

Wir wollen untersuchen, ob der Prozess des *Alternativen Spleißens* anhand von *Exon Skipping Events* ($\mathbf{\Psi}$) rein zufällig stattfindet, oder von regulatorischen Mechanismen gesteuert wird.

- $\mathbf{H_0}$:= *Exon Skipping Events* finden zufällig statt.
- $\mathbf{H_1}$:= *Exon Skipping Events* sind nicht reiner Zufall.

Dafür nehmen wir an, dass die Anzahl an *IRC* Binomial Verteilt ist:

$$P(i, N|p) = \binom{N}{i} p^i (1-p)^{N-i}.$$

Hierbei beschreibt p die Wahrscheinlichkeit, dass ein einzelnes *Exon* als *Inclusion Read* klassifiziert wird, während N die Anzahl an *Reads* darstellt, die für das momentane *Exon* einen Informationsgehalt haben. Zudem gehen wir davon aus, dass jedes Transkript mindestens einen informativen *Read* pro Exon hat.

Für den Test wurden die zehn Ausgabedateien der *JAR* in zwei Gruppen unterteilt (siehe Tabelle 1):

Tabelle 1 — Einteilung der Samples in zwei Gruppen

Ausgabedatei i	1	2	3	4	5	6	7	8	9	10
Gruppe g_i	1	1	1	1	1	2	2	2	2	2
IRC	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}
Gesamt Anzahl	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9	N_{10}
$\mathbf{\Psi_{reduced}} \sim (p_0)$	p_0	p_0	p_0	p_0	p_0	p_0	p_0	p_0	p_0	p_0
$\mathbf{\Psi_{full}} \sim (p_1, p_2)$	p_1	p_1	p_1	p_1	p_1	p_2	p_2	p_2	p_2	p_2

Dabei simulieren wir zwei *Modellen*:

- $\mathbf{\Psi_{reduced}} \sim (p_0)$: Hat einen einzigen Parameter p_0 für alle Gruppen.
- $\mathbf{\Psi_{full}} \sim (p_1, p_2)$: Besitzt zwei verschiedenen Parameter.

Die *Likelihood* kann nun mit der *Maximum Likelihood Estimation* Methode abgeschätzt werden.

Die *Log-Likelihood Funktionen* lasses sich wie folgt definieren:

- $\mathcal{L}_{\mathbf{reduced}}(p_0) := \log\left(\prod_{j=1}^{10} P(i_j, N_j|p_0)\right)$
- $\mathcal{L}_{\mathbf{full}}(p_1, p_2) := \log\left(\prod_{j=1}^{10} P(i_j, N_j|p_{g_j})\right)$

Indem man die jeweilige Ableitung von $\mathcal{L}$ gleich null setzt, erhält man die *MLE* Schätzer $\hat{p}_0$ und $\hat{p}_1, \hat{p}_2$:

$$
\begin{aligned}
\mathcal{L}(\hat{p}_0) &= \log\left(\prod_{j=1}^{10} \binom{N}{i_j} p_0^{i_j}(1-p_0)^{N_j - i_j}\right) \\
\mathcal{L}(\hat{p}_0) &= \sum_{j=i}^{10} \log\left(\binom{N_j}{i_j}\right) + \log(p_0)\sum_{j=1}^{10} i_j + \log(1-p_0)\sum_{j=1}^{10}(N_j - i_j) \\
\mathcal{L}'(\hat{p}_0) &= \frac{\sum_{j=1}^{10} i_j}{p_0} - \frac{\sum_{j=1}^{10}(N_j - i_j)}{1-p_0} \\
0 &= \frac{\sum_{j=1}^{10} i_j}{p_0} - \frac{\sum_{j=1}^{10}(N_j - i_j)}{1-p_0} \\
\hat{p}_0 &= \frac{\sum_{j=1}^{10} i_j}{\sum_{j=1}^{10} N_j}
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{L}(\hat{p}_1, \hat{p}_2) &= \log\left(\prod_{j=1}^{10} \binom{N}{i_j} p_{g_i}^{i_j}(1-p_{g_i})^{N_j - i_j}\right) \\
\iff \mathcal{L}(\hat{p}_1, \hat{p}_2) &= \mathcal{L}_{\mathbf{Gruppe_1}}(p_1) + \mathcal{L}_{\mathbf{Gruppe_2}}(p_2) \\
\mathcal{L}_{\mathbf{Gruppe_1}}'(\hat{p}_1) &= \frac{\sum_{j\in Gruppe_1} i_j}{p_1} - \frac{\sum_{j\in Gruppe_1}(N_j - i_j)}{1-p_1} \\
\iff \hat{p}_1 &= \frac{\sum_{j\in Gruppe_1} i_j}{\sum_{j\in Gruppe_1} N_j} \\
\mathcal{L}_{\mathbf{Gruppe_2}}'(\hat{p}_2) &= \dots \\
\hat{p}_2 &= \frac{\sum_{j\in Gruppe_2} i_j}{\sum_{j\in Gruppe_2} N_j}
\end{aligned}
$$

Die *Likelihood Schätzer* können nun verwendet werde, um $\mathcal{L}_{\mathbf{reduced}}(\hat{p}_0)$ und $\mathcal{L}_{\mathbf{full}}(\hat{p}_1, \hat{p}_2)$ zu berechnen. Zudem bestimmen wir jeweils die *Likelihood Ratio Statistik (LRS)*:

$$LRS := -2 \log \left(\frac{\mathcal{L}_{reduced}}{\mathcal{L}_{full}} \right).$$

Dabei gilt:

$$LRS \sim \chi^2.$$

Somit können wir für jeden der berechneten *PSI* Werte einen p Wert berechnen. Zudem wurde die *Benjamini Hochburg* Methode (Benjamini and Hochberg 1995) verwendet, um die p Werte anzupassen, da multiple Hypothesen Tests durchgeführt wurden.

Das *R* Skript *LRS.R* implementiert die Berechnung der *LRS* und p Werte und schreibt die Resultate in eine Ausgabedatei:

gene	exon	p0	p1	p2	łlreduced	łlfull	łrs	pvalue	padj
id	start-stop	0.64	0.64	0.65	-25.86	-25.82	0.07	0.78	0.9
...	...	...	...	...	...	...	...	...	...

3 — Komplexität und Korrektheit

3.1. Komplexität

Das entwickelte Java Programm benutzt für das Einlesen der *GTF*-Datei, dem Bestimmen der *Exon Skipping Events* und dem *Mapping* der *Reads* dieselbe Komplexität wie die Programme aus Report 1 und 3. Das einzige was an zusätzlichem Aufwand anfällt ist die Erstellung der Intervallbäume pro Gen (A) und das anschließende Zählen der *IRC* und *ERC* (B).

(A) Erstellen der Intervallbäume:

Sei $\tilde{G}$ die Menge aller *Gene* aus der vorliegenden *Annotation* und $\tilde{R}$ die Menge aller *Reads* aus dem Sequenzierexperiment. Dann ist $G \subseteq \tilde{G}$ die Teilmenge an *Genen*, auf welche es ein *Mapping* von *Reads* gibt. Jedes *Gen* $g_i \in G$ hat also eine Teilmenge R_i an alignierten *Reads* aus $\tilde{R}$. Für jedes Gen g_i werden alle zugehörigen *Reads* R_g in die Intervallbäume A und B eingefügt. Das Einfügen in einen Intervallbaum hat ein logarithmisches Kostenmaß: $\mathcal{O}(\log n)$,

wobei n die Gesamtanzahl an Intervallen ist, welche sich bereits in dem Baum befinden. Für das konsekutive Einfügen von *Reads* aus R_g in A ergibt sich folgende, von oben abgeschätzte Laufzeit:

$$\begin{aligned} 1 + \log(1) + \log(2) + \cdots + \log(|R_g| - 1) &< \sum^{|R_g|} \log(|R_g|) \\ &\in \mathcal{O}\Big(|R_g| \cdot \log(|R_g|)\Big) \end{aligned}$$

Das gleiche gilt für die *Gaps*, welche in den zweiten Intervallbaum eingefügt werden:

$$\begin{aligned} 1 + \log(1) + \log(2) + \cdots + \log(R_i/2 + c) &< \sum^{|R_g|/2+c} \log(|R_g|/2 + c) \\ &\in \mathcal{O}\Big((|R_g|/2 + c) \cdot \log(|R_g|/2 + c)\Big) \\ &\in \mathcal{O}\Big((|R_g| + c) \cdot \log(|R_g| + c)\Big) \end{aligned}$$

Da es sich um *paired end* Daten handelt (Abbildung 1), gibt es zwischen jedem *Read Pair* einen *Gap* (also insgesamt $|R_g|/2$) und zusätzlich c viele *Gaps* zwischen den jeweiligen *Alignment-Blocks* der *Reads*.

Da wir insgesamt $|G|$ viele Gene haben also:

$$\mathcal{O}\left(|G| \cdot \Big(|R_{\tilde{g}}| \cdot \log(|R_{\tilde{g}}|) + (|R_{\tilde{g}}| + c) \cdot \log(|R_{\tilde{g}}| + c)\Big)\right)$$

, wobei

$$\tilde{g} := \underset{R_g \in R, R_g \neq R_{\tilde{g}}}{\forall} R_g \leq R_{\tilde{g}}$$

das Gen mit den meisten alignierten *Reads* ist.

(B) Ermittlung der *IRC* und *ERC*:

Eine Suchoperation in einem Intervallbaum hat ebenfalls eine Komplexität von $\mathcal{O}(\log n)$. Nach **Schritt (A)** befinden sich in den Bäumen A, B von einem Gen $g \in G$ genau $|R_i|$ und $|R_i|/2 + c$ viele *Reads*. Jedes $g \in G$ besitzt e_g viele *Exon Skipping Events*. So muss $|e_g|$ oft in jeweils beiden

Bäumen eine Suche getätigt werden. Pro Gen also:

$$|e_g| \cdot \log(R_{\tilde{g}}) + |e_g| \cdot \log(R_{\tilde{g}}/2 + c) = |e_g| \cdot (\log(R_{\tilde{g}}) + \log(R_{\tilde{g}} + c) \in \mathcal{O}\Big(|e_g| \cdot \log(R_{\tilde{g}} + c)\Big).$$

3.2. Korrektheit

Der Ansatz mit den Intervallbäumen ist Korrekt. In den Intervallbäumen werden nur *Reads* abgespeichert, die genau auf ein *Transkript* passen. Somit werden schon mal alle *Reads* ohne eindeutigen Informationsgehalt ignoriert. Zusätzlich werden zu jeder hinzugefügten Region (in beiden Bäumen), die *Ursprungs ID* (in diesem Fall die Transkript ID) und die *Read ID* gespeichert. Da der Intervallbaum korrekt implementiert ist, liefert diese entweder alle Intervalle, die von der übersprungenen *CDS* beinhaltet werden (Baum A), oder alle *Gaps*, welche die *CDS* überspannen (Baum B). Die Anzahl an einzigartigen *Read IDs* beider Abfragen ergibt dann die jeweiligen *IRC* und *ERC*. Der einzige Sonderfall ergibt sich, wenn eine Region r aus dem Baum B dieselbe *Transkript ID* hat wie das der übersprungenen *CDS*. In diesem Fall impliziert r dann keinen *ERC* sonder einen *IRC*, denn r wurde schließlich eindeutig auf das *Transkript* der *CDS* aligniert.

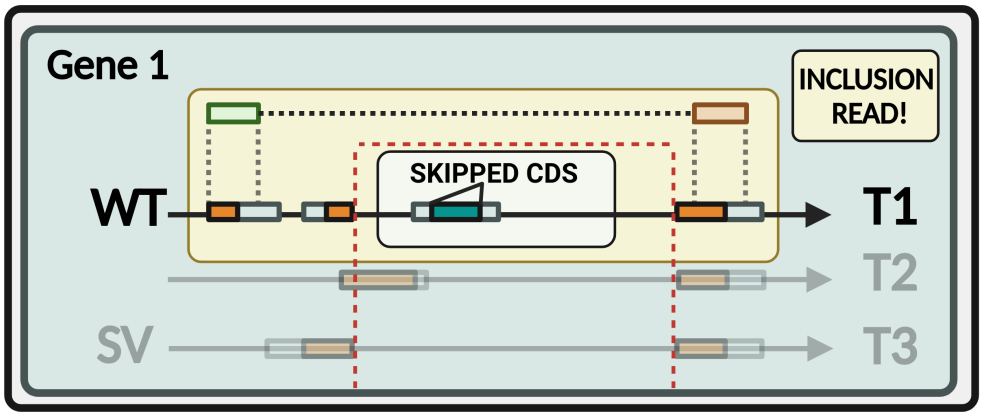

Abbildung 2 — Sonderfall bei der Berechnung von *ERC*. Der *Read* passt hier nur auf T1 und obwohl sich die übersprungene *CDS* in der Lücke zwischen *Forward* und *Reverse Read* befindet, zählt dieser *Read* als *IRC*, denn er kann nur durch T1 entstanden sein. Die Abbildung wurde mit BioRender 2024 erstellt

4 — Ergebnisse

4.1. PSI Werte

4.2. Vergleich mit DEXSeq

4.3. Laufzeit

Referenzen

Anders, Simon, Alejandro Reyes, and Wolfgang Huber. 2012. "Detecting differential usage of exons from RNA-seq data." *Genome Research* 22:4025. https://doi.org/10.1101/gr.133744.111. (Cited on page 1).

Benjamini, Yoav, and Yosef Hochberg. 1995. "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing." *Journal of the Royal Statistical Society: Series B (Methodological)* 57 (1): 289–300. https://doi.org/https://doi.org/10.1111/j.2517-6161.1995.tb02031.x. eprint: https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/j.2517-6161.1995.tb02031.x. https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1995.tb02031.x. (Cited on page 8).

BioRender. 2024. *BioRender - Biological Figure Creation Tool.* https://BioRender.com. Accessed: 2024-12-09. (Cited on pages 4, 10).