Probability Generating Functions for interaction counting

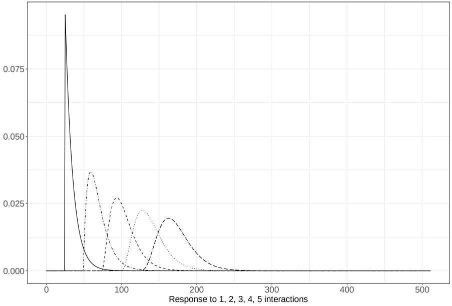
Vladik Balagura
LLR, CNRS, Ecole Polytechnique, Institut Polytechnique de Paris

21 Oct 2020

Response to N interactions

Let $p(N)$ = luminometer spectrum - probability to have luminometer value N (eg. N hits).

Example: solid curve - response to 1 interaction.



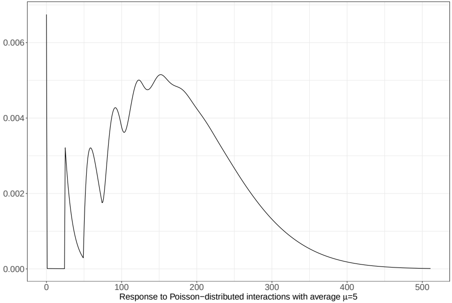
Response to N interactions = N convolutions of $p(N)$ with itself.

Its Fourier transform: $(p^F)^N$, where p^F - Fourier transform of $p(N)$.

Response to μ Poisson-distributed interactions

$\mathcal{P}$oisson probability $\mathcal{P}(n) = \frac{\mu^n}{n!}e^{-\mu}$, so Fourier transform of response to μ Poisson-distributed interactions:

$$\mathcal{P}(p)^F = \sum_n \frac{(p^F)^n \mu^n}{n!} e^{-\mu} = e^{-\mu(p^F-1)}$$



Example for $\mu = 5$. Note δ-function at zero-bin, $\mathcal{P}(0) = e^{-\mu}$. I've realized this formalism $\sim$ 15 years ago when fitting SiPM spectra. Published as Appendix in NIM A 564 (2006) p.590 ("Study of scintillator strip with wavelength shifting fiber and silicon photomultiplier").

Equally well can be applied to any luminometer.

Probability Generating Functions (PGF)

$$G_p(z) = \overline{z^N} = E_p[z^N] = \sum_{N=0}^{M-1} p(N) \cdot z^N$$

Fourier: $p^F(K) = \sum_{N=0}^{M-1} p(N) e^{-2\pi i \cdot N \cdot \frac{K}{M}}$, inverse: $p(N) = \frac{1}{M} \sum_{K=0}^{M-1} p^F(K) e^{2\pi i \cdot K \cdot \frac{N}{M}}$

Fourier transform is a special case of PGF when $z = e^{2\pi i\, K/M}$. Like Fourier transforms:

generating function of discrete convolution of $p(N), q(N) = p * q$ is the product $G(p) \cdot G(q)$:

$$\sum_{N=0}^{M-1} p(N) \cdot z^N \times \sum_{K=0}^{M-1} q(K) \cdot z^K = \sum_{L=0}^{M-1} \left(\sum_{N,M,N+M=L} p(N) q(K) \right) \cdot z^L = \sum_{L=0}^{M-1} (p * q)(L) \cdot z^L.$$

Therefore, PGF of response to μ Poisson-distributed interactions:

$$G_{\mathcal{P}(p)}(z) = \sum_n \frac{(G_p)^n \mu^n}{n!} e^{-\mu} = e^{-\mu [G_p(z) - 1]}$$

So, $G_{\mathcal{P}(p)}(z) = e^{-\mu[G_p(z)-1]}$. Can this be used in practice?

Let's denote incoming per-event detector measurements by $N_1, N_2, N_3 \ldots$.
Take any (even complex) number z and calculate average z^{N_i}

$$G_{\mathcal{P}(p)}(z) = \overline{z^{N_i}} = \frac{\sum_i z^{N_i}}{N_{ev.}},$$

where $N_{ev.}$ – number of events. Then,

$$\log(G_{\mathcal{P}(p)}(z)) = -\mu \log(G_p(z) - 1) \propto \mu \propto \text{Luminosity}$$

New method of luminosity measurement in addition to known "average" and "logZero":

"Average", "logZero" and PGF algorithms

	Accumulate	Lumi
Average	$N_1 + N_2 + \ldots$	$\propto \frac{\sum N_i}{n}$
LogZero	$1 + 0 + \ldots$	$\propto -\log \frac{\sum(N_i{==}0)}{n}$
G(z)	$z^{N_1} + z^{N_2} + \ldots$	$\propto \log(\frac{\sum z^{N_i}}{n})$

"Average" and "logZero" are special cases of PGF method!

Take $z = 1 + \epsilon$ and the limit $\epsilon \to 0$:

$$\lim_{\epsilon\to 0} G(1+\epsilon) = \lim_{\epsilon\to 0} \log\left(\frac{\sum_i (1+\epsilon)^{N_i}}{N_{ev.}}\right) = \lim_{\epsilon\to 0} \log\left(1 + \sum_i \frac{\epsilon N_i}{N_{ev.}}\right) = \epsilon \overline{N}.$$

So,

$$\text{Luminosity} \propto \overline{N} = \frac{1}{\epsilon} \lim_{\epsilon\to 0} G(1+\epsilon).$$

$z = \epsilon \to 0$ reproduces "logZero": $\epsilon^0 = 1$, $\epsilon_i^N \to 0$ for $N_i > 0$, so only zero-bin $N_i = 0$ matters and

$$-\lim_{\epsilon\to 0} G(\epsilon) = -\lim_{\epsilon\to 0} \log\left(\frac{\sum_i \epsilon^{N_i}}{N_{ev.}}\right) = -\log\left(\frac{N_{ev.}^{N=0}}{N_{ev.}}\right) \propto \text{Luminosity}.$$

Which method is better? "Average"?

(+) Simple and linear

(-) Relies on

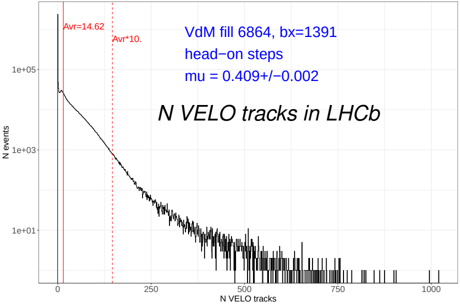
- long-term stability of calibration constant and
- linearity in full range (bias from high pile-up evs. or saturation: N triggered $\leq$ total N chan.)

Note, bias from badly reconstructed "busy" events is enhanced proportionally to N_i:

$$\overline{N} = \frac{\sum_{ev.} N_i}{N_{ev.}} = \sum_N (p_N \cdot N)$$

(many tracks in busy ev. can be biased at once).

$\rightarrow$ To be safe, long time ago in LHCb we've decided to use "logZero" instead.



"LogZero" method, $\mu = -\log(\mathcal{P}_0)$

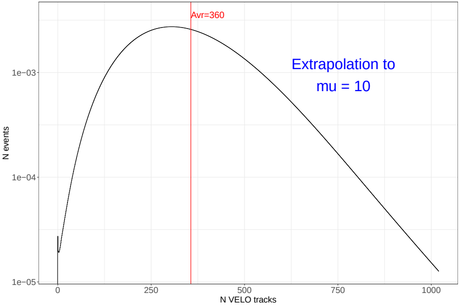
(+) Insensitive to reconstruction in busy events: all classified as non-empty

(-) Does not work at high μ: $\mathcal{P}_0 = e^{-\mu}$ too small

- "LogZero" worked fine at LHCb in Runs 1-2 ($\mu \approx 1$).
- But with $\mu \sim 5 - 10$ in Runs 3-4 in large acceptance luminometers (eg. Velo):

μ	$\mathcal{P}(0)$
5	$7 \cdot 10^{-3}$
10	$5 \cdot 10^{-5}$

– too small (and too sensitive to μ variations).



Let's use PGF!

Let's take $0 < z < 1$: then in

$$G(z) = \log \left(\frac{\sum_{ev.} z^{N_i}}{N_{ev.}} \right) \propto \text{Luminosity}$$

higher N-bins exponentially vanish as z^{N_i}. This automatically suppresses bias in "busy" events!

z can be optimized for a given N-spectrum. Eg.

$$z = 2^{-1/N_0} = \sqrt[N_0]{0.5}$$

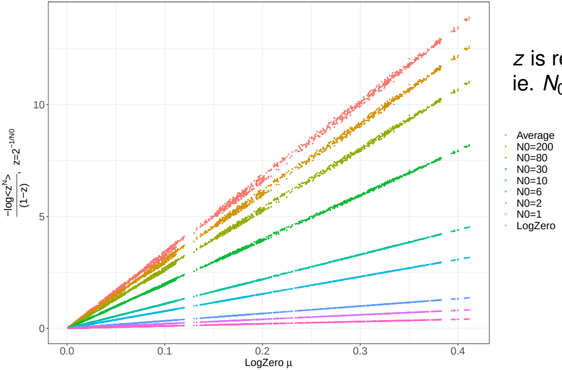
suppresses N_0-th bin by $z_0^N = 2^{-1}$, $2N_0$-th by 2^{-2} etc., and N_0 can be tuned.

This is a way to go for LHCb in Runs 3, 4!

Example from van der Meer scan with varying μ

	method	line	
	"average"	upper: "avr." vs. "log0"	$z \to 1$
Vertical	PGF	intermediate	$0 < z < 1$
	"LogZero"	botoom at 45°	$z \to 0$
Horizontal	"logZero"		

Good linearity everywhere as expected.



z is represented as $z = \sqrt[N_0]{0.5}$, ie. N_0th bin is suppressed by 2

To have continuous transition "logZero" $\to$ PGF $\to$ "average", $\log G(z) \propto L$ is normalized by $-(1-z)^{-1}$.

Summary

- New unbiased method of interaction counting is proposed. Backed by sound math: PGF.
- Will be deployed at LHCb for next Runs, can be used in other experiments, eg. to suppress high pile-up events in ATLAS, CMS.
- z-variation in the range $0 < z < 1$ performs continuous transition between "logZero" and "average". New degree of freedom for
  - optimizing luminometers,
  - cross-checking results using the same data.
- Can be used universally with any luminometer, discrete ($G(z) = \sum_N p_N z^N$) or continuous ($\int p(x) z^x dx$).
- Can also be applied for measuring other "extensive" (or additive) variables (aka luminosity) via experimental spectra.
- Exponential response $e^{\alpha N}$ (instead of αN) can be implemented already at hardware level. Then, instead of $\overline{N}$, one determines $\log(\overline{e^{\alpha N}})$.
- Method works for any z. There might be specific cases when eg. negative, $z > 1$ or complex z might be useful.