desietc Documentation

Release 0.1.0.dev35

DESI

Nov 17, 2017

Contents

CHAPTER 1

Introduction

This is the documentation for DESI ETC.

CHAPTER 2

Contents

2.1 Change Log

2.1.1 0.2.0 (unreleased)

- Update to latest desitemplate boilerplate.

2.1.2 0.1.0 (2017-10-29)

- Initial tag of code completed Aug 2015, used for DESI-1100.

2.2 desietc API

High-level calculator for online exposure-time forecasts.

class desietc.calculator.**Calculator**(*alpha*, *dalpha*, *beta*, *dbeta*, *sig0*, *dsig0*, *tcorr_sig*, *bg0*, *dbg0*, *tcorr_bg*, *t0*, *snr_goal*, *dtmax=4000.0*, *npred=401*)

Create a new calculator object for a single spectrograph exposure.

Model the time-evolution of independent signal and background rate estimators, and use models to forecast the currently acheived signal-to- noise ratio (SNR) and the remaining exposure time.

Register updates to the estimated signal and background rates using *update_signal()* and *update_background()*, respectively. The model is initialized with an initial guess at these rates which is gradually replaced by actual rate updates. The signal and background rate estimates are assumed to be statistically independent.

Use *will_timeout()*, *get_snr_now()* and *get_remaining()* to query the latest models.

The target SNR is calculated as $S/\sqrt{(S+B)}$ where $S = \alpha s$ and $B = \beta b$ are calibrated versions of the estimates s and b passed to *update_signal()* and *update_background()*. The dimensions of a, b, α, and β are

arbitrary, but the combinations $S = \alpha s$ and $B = \beta b$ must be dimensionless and scaled appropriately to have counting statistics.

The calibration constants α, and β can have associated errors, σ_α and σ_β that are propagated to SNR estimates. In general, the calibration depends on the program (DARK, GRAY, BRIGHT) and can be refined by comparing this object's predictions with the pipeline outputs from previous spetrograph exposures.

Parameters

- **alpha** (*float*) – Constant conversion factor from raw signal rate to fiducial target signal rate. Must be > 0.
- **dalpha** (*float*) – Error on alpha. Must be > 0.
- **beta** (*float*) – Constant conversion factor from raw signal rate to fiducial target signal rate. Must be > 0.
- **dbeta** (*float*) – Error on beta. Must be > 0.
- **sig0** (*float*) – Prior on raw (uncalibrated) signal rate. Must be > 0.
- **dsig0** (*float*) – One sigma error on `sig0`. Must be > 0.
- **tcorr_sig** (*float*) – Correlation time for changes in signal rate. Acts as a prior on how rapidly the signal rate changes. Must be > 0.
- **bg0** (*float*) – Prior on raw (uncalibrated) background rate. Must be > 0.
- **dbg0** (*float*) – One sigma error on `bg0`. Must be > 0.
- **tcorr_bg** (*float*) – Correlation time for changes in background rate. Acts as a prior on how rapidly the background rate changes. Must be > 0.
- **t0** (*float*) – Timestamp for when shutter was opened for the current exposure, in units of seconds with an arbitrary origin.
- **snr_goal** (*float*) – Value of calibrated SNR that we ideally want to achieve. Must be > 0.
- **dtmax** (*float*) – Maximum allowed total exposure time in seconds. Must be > 0.
- **npred** (*int*) – Number of equally spaced times spanning [0, dtmax] where predictions are calculated internally.

_eval_snr(*t*, *S*, *B*)

Evaluate the SNR for calibrated rates S and B at increasing times t.

Automatically broadcasts over input arrays whose last index corresponds to time.

Parameters

- **t** (*array*) – 1D array of N increasing times in seconds.
- **S** (*array*) – Array with shape (...,N) of calibrated signal rates in counts per second at each time.
- **B** (*array*) – Array with shape (...,N) of calibrated background rates in counts per second at each time.

Returns Array with shape (...,N) of cummulative signal-to-noise ratios at each time calculated as $S/np.sqrt(S+B)$. Any time when $S + B \leq 0$ will have SNR = 0.

Return type array

_update_model(*dt*, *rate*, *drate*, *rate0*, *drate0*, *tcorr*)
Internal method to update a rate model.

This method is used by both *update_signal()* and *update_background()* to update their respective models.

Call `update_snr()` after calling this method to calculate the updated SNR.

Parameters

- **dt** (*array*) – 1D array of N elapsed times in seconds since `self.t0`, not necessarily in increasing order. An empty array (N=0) is allowed.
- **rate** (*array*) – 1D array of N raw (uncalibrated) rate estimates corresponding to each `dt` value. Must be > 0.
- **drate** (*array*) – 1D array of N 1-sigma errors on each `rate` value. Must be > 0.
- **rate0** (*float*) – Prior on the raw (uncalibrated) rate that is combined with the rate estimates to obtain a posterior estimate of the rate. Must be > 0.
- **drate0** (*float*) – 1-sigma error on the prior value `rate0`. Must be > 0.
- **tcorr** (*float*) – Correlation time in seconds for changes in the raw rate. Acts as a prior on how rapidly the rate changes. Must be > 0.

Returns Tuple (pred, dpred, gp) where pred and dpred are arrays of predicted rates and their 1-sigma errors for each elapsed time in `self.dt_pred`, and `gp` is the updated Gaussian process model.

Return type tuple

_update_snr()
Internal method to update SNR forecast.

Uses the the most recent updates to the signal and background models to forecast the calibrated SNR out to `dtmax`. This forecast is then used to estimate the current SNR and the time remaining until `snr_goal` is achieved.

Sets the flag `attr:timeout` if the updated model indicates that this exposure will not complete before `dtmax` is reached.

get_remaining(*tnow*)
Forecast the remaining exposure time in seconds.

Based on all signal and background rate updates recorded so far.

Parameters **tnow** (*float*) – Current time used for forecasting. Must be >= t0.

Returns Remaining time in seconds, which might be < 0 if the goal SNR has already been achieved.

Return type float

get_samples(*dt*, *nsamples*, *gen=None*)
Generate random samples of our signal and background rate models.

Parameters

- **dt** (*array of floats*) – Array of times where models should be sampled. Times are in seconds relative to the exposure start. Must all be between 0 and dtmax.
- **nsamples** (*int*) – Number of independent samples to generate. Must be > 0.
- **gen** (*numpy.random.RandomState or None*) – Use the specified random number generator for reproducible results.

Returns Tuple (S, B, snr) of calibrated signal and background rate samples, and the corresponding SNR values. Each is an array of floats with dimensions (nsamples, len(dt)).

Return type tuple

get_snr_now(*tnow*, *CL=0.6827*, *nsamples=1000*, *gen=None*)

Estimate the current integrated SNR.

Based on all signal and background rate updates recorded so far.

This calculation is relatively expensive since it requires generating random samples from the signal and background rate models.

Parameters

- **tnow** (*float*) – Current time used for forecasting. Must be between t0, t0+dtmax.
- **CL** (*float*) – Confidence level to use for the returned range. Must be 0-1.
- **nsamples** (*int*) – Number of random samples of the signal and background rate models to generate. A larger number gives more accurate ranges but takes longer to run. Must be > 0.
- **gen** (*numpy.random.RandomState or None*) – Use the specified random number generator for reproducible results.

Returns Tuple (lo, hi) containing the true integrated SNR with posterior probability CL.

Return type tuple

update_background(*tbg*, *bg*, *dbg*)

Update the background estimate.

Can be called with timestamps `tbg` in any order.

Parameters

- **tbg** (*float*) – Timestamp in seconds for this background rate estimate, using the same (arbitrary) origin as the `t0` passed to the constructor. The elapsed time `tbg - t0` must be >= 0.
- **bg** (*float*) – Estimated raw (uncalibrated) background rate at `tbg`. Must be > 0.
- **dbg** (*float*) – Estimated 1-sigma error on the value of `bg`. Must be > 0.

update_signal(*tsig*, *sig*, *dsig*)

Update the signal estimate.

Can be called with timestamps `tsig` in any order.

Parameters

- **tsig** (*float*) – Timestamp in seconds for this signal rate estimate, using the same (arbitrary) origin as the `t0` passed to the constructor. The elapsed time `tsig - t0` must be >= 0.
- **sig** (*float*) – Estimated raw (uncalibrated) signal rate at `tsig`. Must be > 0.
- **dsig** (*float*) – Estimated 1-sigma error on the value of `sig`. Must be > 0.

will_timeout()

Is this exposure expected to time out before reaching its goal SNR?

Based on all signal and background rate updates recorded so far.

Returns True if the current exposure is not expected to achieve `snr_goal` with an exposure duration less than `dtmax`.

Return type bool

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

d

Index

Symbols

C

D

G

U

W