GammaLib - Feature #3882

Add polarization to interface

10/19/2021 04:10 PM - Knödlseder Jürgen

Status:	In Progress	**Start date:**	10/19/2021
Priority:	Normal	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	10%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

Polarization should be added to the general obs interfaces so that data for instruments that can measure polarisation can also be analysed. Polarisation should be implemented by a dedicated GPolarisation class that specifies the true polarisation of a photon.

History

#1 - 10/20/2021 11:13 AM - Knödlseder Jürgen

- Status changed from New to In Progress

- Assigned To set to Knödlseder Jürgen

- % Done changed from 0 to 10

I added a class GPolarization that will implement polarization information. So far the class does not contain any information. The objective so far is to adapt all interfaces so that polarization can be handled in the future.

#2 - 10/20/2021 02:58 PM - Knödlseder Jürgen

I implemented the following changes to add polarization to the GammaLib interface

```
virtual const GPolarization& GEvent::polarization(void) const = 0;
virtual double GModel::npred(const GEnergy& obsEng, const GTime& obsTime, const GPolarization& obsPol, const GObservation& obs) const = 0;
virtual double GModelSpatial::flux(const GSkyRegion& region, const GEnergy& srcEng  = GEnergy(), const GTime& srcTime = GTime(), const GPolarization& srcPol = GPolarization()) const;
GPhoton(const GSkyDir& dir, const GEnergy& energy, const GTime& time, const GPolarization& polarization, const int& mc_id = -1);
const GPolarization& GPhoton::polarization(void) const;
void GPhoton::polarization(const GPolarization& polarization);
virtual double GResponse::nroi(const GModelSky& model, const GEnergy& obsEng, const GTime& obsTime, const GPolarization& obsPol,  const GObservation&  obs) const = 0;
GSource(const std::string& name, GModelSpatial* model, const GEnergy& energy, const GTime& time, const GPolarization& polarization);
const GPolarization& GSource::polarization(void) const;
void GSource::polarization(const GPolarization& polarization);
```

Note that these changes assume that polarization in physical and data space have the same parametrisation, which will however not be the case in general. Hence a new class

```
GInstPol
```

should be introduced that is used in

```
virtual const GInstPol& GEvent::polarization(void) const = 0;
virtual double GResponse::nroi(const GModelSky& model, const GEnergy& obsEng, const GTime& obsTime, const GInstPol& obsPol,  const GObservation&  obs) const = 0;
```

and instrument specific derived classes should then implement any instrument specific polarisation.

I'm however wondering whether such a general scheme is indeed appropriate since most instruments will not be able to measure polarisation.

#3 - 10/20/2021 04:04 PM - Knödlseder Jürgen

- Target version deleted (2.0.0)

Let's put this issue on hold unless we have a better view on how to implement polarisation.