```
#include <vector>

#include "boost/smart_ptr/shared_ptr.hpp"

#include "gridpack/include/gridpack.hpp"

#include "pf_factory.hpp"



namespace gridpack {
namespace powerflow {

// Powerflow factory class implementations
```

Basic constructor
@param network: network associated with factory

```
PFFactory::PFFactory(PFFactory::NetworkPtr network)
  : gridpack::factory::BaseFactory<PFNetwork>(network)
{
  p_network = network;
}
```

Basic destructor

```
gridpack::powerflow::PFFactory::~PFFactory()
{
}
```

The $\overline{\overline{Y}}$-matrix relates the vector of currents $\overline{I}$ to the voltages $\overline{V}$ via $\overline{I} = \overline{\overline{Y}} \cdot \overline{V}$

Create the admittance (Y-Bus) matrix.

```
void gridpack::powerflow::PFFactory::setYBus(void)
{
  int numBus = p_network->numBuses();
  int numBranch = p_network->numBranches();
  int i;

  // Invoke setYBus method on all branch objects
  for (i=0; i<numBranch; i++) {
    p_network->getBranch(i).get()->setYBus();
  }

  // Invoke setYBus method on all bus objects
  for (i=0; i<numBus; i++) {
    p_network->getBus(i).get()->setYBus();
  }

}
```

Make SBus vector

```
void gridpack::powerflow::PFFactory::setSBus(void)
{
  int numBus = p_network->numBuses();
  int i;

  // Invoke setSBus method on all bus objects
  for (i=0; i<numBus; i++) {
    dynamic_cast<PFBus*>(p_network->getBus(i).get())->setSBus();
  }
}
```

```
} // namespace powerflow
} // namespace gridpack
```