Rate and Range checks:

Preflight checks and errors in PX4:

Preflight fail: EKF HGT ERROR:
- This error is produced when the IMU and height measurement data are inconsistent.
- Perform an accelerometer and gyroscope calibration and restart the vehicle. If the error persists, check the height sensor data for problems.
- The check is controlled by the COM_ARM_EKF_HGT parameter.

Preflight fail: EKF HORIZ POS ERROR:
- This error is produced when the IMU and position measurement data (either GPS or external vision) are inconsistent.
- Check the position sensor data for un-realistic data jumps. If data quality looks OK, perform an accel and gyro calibration and restart the vehicle.
- The check is controlled by the COM_ARM_EKF_POS parameter.

Preflight fail: EKF YAW ERROR:
- This error is produced when the yaw angle estimated using gyro data and the yaw angle from the magnetometer or external vision system are inconsistent.
- Check the IMU data for large yaw rate offsets and check the magnetometer alignment and calibration.
- The check is controlled by the COM_ARM_EKF_YAW parameter
- The default value of 0.5 allows the differences between the navigation yaw angle and magnetic yaw angle (magnetometer or external vision) to be no more than 50% of the maximum tolerated by the EKF and provides some margin for error increase when flight commences.
- It can fail if the yaw gyro has a large offset or if the vehicle is moved or rotated in the presence of a bad magnetic interference or magnetometer calibration.
Preflight fail: EKF HIGH IMU ACCEL BIAS:
- This error is produced when the IMU accelerometer bias estimated by the EKF is excessive.
- The check is controlled by the COM_ARM_EKF_AB parameter.

Preflight fail: EKF HIGH IMU GYRO BIAS:
- This error is produced when the IMU gyroscope bias estimated by the EKF is excessive.
- The check is controlled by the COM_ARM_EKF_GB parameter.

Preflight fail: EKF VEL ERROR:
- This error is produced when the IMU and GPS velocity measurement data are inconsistent.
- Check the GPS velocity data for un-realistic data jumps. If GPS quality looks OK, perform an accel and gyro calibration and restart the vehicle.
- The check is controlled by the COM_ARM_EKF_VEL parameter.

Preflight fail: EKF INTERNAL CHECKS:
- This error message is generated if the innovation magnitudes of either the horizontal GPS velocity, magnetic yaw, vertical GPS velocity or vertical position sensor (Baro by default but could be rangefinder or GPS if non-standard parameters are being used) are excessive. Innovations are the difference between the value predicted by the inertial navigation calculation and measured by the sensor.
Users should check the innovation levels in the log file to determine the cause. These can be found under the `ekf2_innovations` message. Common problems/solutions include:

- IMU drift on warmup. May be resolved by restarting the autopilot. May require an IMU accel and gyro calibration.
- Adjacent magnetic interference combined with vehicle movement. Resolve by moving vehicle and waiting or re-powering.
- Bad magnetometer calibration combined with vehicle movement. Resolve by recalibrating.
- Initial shock or rapid movement on startup that caused a bad inertial nav solution. Resolve by restarting the vehicle and minimising movement for the first 5 seconds.

Preflight fail: COMPASS SENSORS INCONSISTENT - CHECK CALIBRATION:

- This error message is produced when the difference in measurements from different compass sensors is too great.
- It indicates bad calibration, orientation or magnetic interference.
- This check only applies to when more than one compass/magnetometer is connected.
- The check is controlled by the `COM_ARM_MAG_ANG` parameter.

Other parameters:

The `COM_ARM_WO_GPS` parameter controls whether or not arming is allowed without a global position estimate.

- (default): Arming is allowed without a position estimate for flight modes that do not require position information (only).
- Arming is allowed only if EKF is providing a global position estimate and EFK GPS quality checks are passing.

CHECKS MATLAB DOES:

**Troubleshoot Signal Range Errors**
Simulink provides a diagnostic named *Simulation range checking*, which you can enable to detect when signals exceed their specified ranges during simulation. When enabled, Simulink compares the signal values that a block outputs with both the specified range (see *Work with Signal Ranges in Blocks*) and the block data type. That is, Simulink performs the following check:

\[
\text{DataTypeMin} \leq \text{MinValue} \leq \text{VALUE} \leq \text{MaxValue} \leq \text{DataTypeMax}
\]

**Identify unconnected lines, input ports, and output ports**

**Check ID:** `mathworks.design.UnconnectedLinesPorts`

Check for unconnected lines or ports.

**Check root model Inport block specifications**

**Check ID:** `mathworks.design.RootInportSpec`

Check that root model Inport blocks fully define dimensions, sample time, and data type.

**Check optimization settings**

**Check ID:** `mathworks.design.OptimizationSettings`

Check for optimizations that can lead to non-optimal code generation and simulation.

**Check diagnostic settings ignored during accelerated model reference simulation**

**Check ID:** `mathworks.design.ModelRefSIMConfigCompliance`

Checks for referenced models for which Simulink changes configuration parameter settings during accelerated simulation.

**Check for parameter tunability information ignored for referenced models**
Check ID: mathworks.design.ParamTunabilityIgnored

Checks if parameter tunability information is included in the Model Parameter Configuration dialog box.

**Check for implicit signal resolution**
Check ID: mathworks.design.ImplicitSignalResolution

Identify models that attempt to resolve named signals and states to Simulink.Signal objects.

**Check for optimal bus virtuality**
Check ID: mathworks.design.OptBusVirtuality

Identify virtual buses that could be made nonvirtual. Making these buses nonvirtual improves generated code efficiency.

**Check for Discrete-Time Integrator blocks with initial condition uncertainty**
Check ID: mathworks.design.DiscreteTimeIntegratorInitCondition

Identify Discrete-Time Integrator blocks with state ports and initial condition ports that are fed by neither an Initial Condition nor a Constant block.

**Check for large number of function arguments from virtual bus across model reference boundary**
Check ID: mathworks.design.CheckVirtualBusAcrossModelReferenceArgs

Checks virtual bus signals that cross model reference boundaries and flags cases where using virtual buses across a model reference boundary increases the number of function arguments significantly.

**Check structure parameter usage with bus signals**
Check ID: mathworks.design.MismatchedBusParams

Identify blocks and Simulink.Signal objects that initialize bus signals by using mismatched structures.

**Check if all simulation outputs are returned as a single Simulink.SimulationOutput object**
Check ID: mathworks.design.CheckSingleSimulationOutput
Use this check to identify if the simulation result is returned as a single `Simulink.SimulationOutput` object.

**Check Rapid accelerator signal logging**

**Check ID:** `mathworks.design.CheckRapidAcceleratorSignalLogging`

When simulating your model in rapid accelerator mode, use this check to find signals logged in your model that are globally disabled. Rapid accelerator mode supports signal logging. Use this check to enable signal logging globally.