Mathematica Code and Usage Method

Here is the function that calculates QFIM. first input argument is density matrix (mat), and second input argument is an array of desired parameters ($variables$)

CODE

```
(* This function calculates QFIM of density matrix "mat" for variables
"variables" *)
    Fisher[mat_,variables_]:=(
    result=Array[0&,{Length[variables],Length[variables]}];
    evals=Simplify[Eigenvalues[mat]];
    evecs=Eigenvectors[mat]//Simplify;
    For[i=1,i<=Length[variables],i++,
    For[j=1,j<=Length[variables],j++,
    firstVar=variables[[i]];
    secondVar=variables[[j]];
    fisher1=fisher2=fisher3=0;
    For[k=1,k<=Length[evals],k++,
    If[evals[[k]]=!=0,
    eval1=evals[[k]];
    evec1=evecs[[k]];
    KetEvec1=Simplify[Transpose[{evec1}]];
    BraEvec1=Simplify[ConjugateTranspose[KetEvec1]];
    norm1=Sqrt[(Simplify[BraEvec1.KetEvec1])[[1,1]]];
    KetEvec1=Simplify[KetEvec1/norm1];
    BraEvec1=Simplify[BraEvec1/norm1];
    BraEvec1Prime1=D[BraEvec1,firstVar];
    KetEvec1Prime1=D[KetEvec1,firstVar];
    BraEvec1Prime2=D[BraEvec1,secondVar];
    KetEvec1Prime2=D[KetEvec1,secondVar];
    eval1Prime1=D[eval1,firstVar];
    eval1Prime2=D[eval1,secondVar];
    fisher1+=eval1Prime1*eval1Prime2/eval1;
    fisher2+=eval1*Re[(BraEvec1Prime1.KetEvec1Prime2)[[1,1]]];
    For[l=1,l<=Length[evals],l++,
    If[evals[[l]]=!=0 ,
    eval2=evals[[l]];
    evec2=evecs[[l]];
    KetEvec2=Simplify[Transpose[{evec2}]];
    BraEvec2=Simplify[ConjugateTranspose[KetEvec2]];
    norm2=Sqrt[(Simplify[BraEvec2.KetEvec2])[[1,1]]];
    KetEvec2=Simplify[KetEvec2/norm2];
    BraEvec2=Simplify[BraEvec2/norm2];
    BraEvec2Prime1=D[BraEvec2,firstVar];
    KetEvec2Prime1=D[KetEvec2,firstVar];
    BraEvec2Prime2=D[BraEvec2,secondVar];
    KetEvec2Prime2=D[KetEvec2,secondVar];
    fisher3+=eval1*eval2*Re[(BraEvec1Prime1.KetEvec2)[[1,1]]
    *(BraEvec2.KetEvec1Prime2 )[[1,1]]]/(eval1+eval2);
    ];];];];
```

```
    Print[variables[[i]],variables[[j]]," done"];
    result[[i,j]]=(fisher1+4*fisher2-8*fisher3)//Simplify;
    ];];
    ClearAll[i,j,k,l,fisher1,fisher2,fisher3,KetEvec1,
    KetEvec1Prime1,KetEvec1Prime2,KetEvec2,KetEvec2Prime1,
    KetEvec2Prime2,BraEvec1,BraEvec1Prime1,BraEvec1Prime2,
    BraEvec2,BraEvec2Prime1,BraEvec2Prime2,evecs,evec1,
    evec2,evals,eval1,eval1Prime1,eval2,eval1Prime2,
    firstVar,secondVar];
    Return[result];);
```

And here is how to use it for Fock state with $n = 1$. At first we define stateket $|\psi\rangle$, and then calculate density matrix ρ.

CODE USAGE EXPLANATION

```
(* one Photon in mach zehnder - there are 3 distinct states *)
(* set global assumptions *)
$Assumptions = {T, φ, γ} ∈ Reals && T >= 0 && T <= 1;
(* Define transmission and reflection coefficients *)
t = T Exp[-I γ];
r = Sqrt[1 - T^2];
(*  |ψ> = a1|100> + a2|010> + a3|001>  *)
a1 = I ( t - Exp[-I φ ])/2;
a2 = -(t + Exp[-I φ ])/2;
a3 = I r Sqrt[2]/2;
ψ = {{a1}, { a2}, { a3}};
ψ // MatrixForm;
ρ = ψ.ConjugateTranspose[ψ] // FullSimplify;
ρ // MatrixForm
Norm[ψ] // FullSimplify;
```

Now we must trace out lost photons

CODE USAGE EXPLANATION

```
(* Tracing over dissipated photons in path |w> *)
ρ[[3, 1]] = ρ[[3, 2]] = ρ[[1, 3]] = ρ[[2, 3]] = 0;
ρ // MatrixForm
```

Now we calculate QFIM and Covariance matrix

CODE USAGE EXPLANATION

```
fisher = Fisher[ρ, {γ, T}];
fisher = fisher // ComplexExpand // Simplify;
fisher // MatrixForm
cov = Inverse[fisher // Simplify] // Simplify;
cov // MatrixForm
```